

128/p115

10018823/018823

JCO3 Rec'd PCT/PTC 19 DEC 2001

1

## DESCRIPTION

### Information Processing Apparatus and Method, Program and Recording Medium

#### Technical Field

This invention relates to an information processing method and apparatus, and to a program. More particularly, it relates to an information processing method and apparatus, a recording medium and a program, in which the information, such as the address information, encoding parameters, transition point information and marks of an I-picture in an AV stream, is recorded as a file.

#### Background Art

Recently, a variety of types of optical discs have been proposed as a recording medium that can be removed from a recording apparatus. These recordable optical discs have been proposed as a large capacity medium of several GBs and are thought to be promising as a medium for recording AV (audio visual) signals, such as video signals. Among the digital AV signal sources (supply sources), to be recorded on this recordable optical disc, there are CS digital satellite broadcast and BS digital broadcast. Additionally, the ground wave television broadcast of the digital system has also been proposed for future use.

The digital video signals, supplied from these sources, are routinely image-compressed under the MPEG (Moving Picture Experts Group) 2 system. In a

recording apparatus, a recording rate proper to the apparatus is set. If digital video signals of the digital broadcast are recorded in the conventional image storage mediums for domestic use, digital video signals are first decoded and subsequently bandwidth-limited for recording. In the case of the digital recording system, including, of course, the MPEG1 Video, MPEG2 video and DV system, digital video signals are first decoded and subsequently re-encoded in accordance with an encoding system for the recording rate proper to the apparatus for subsequent recording.

However, this recording system, in which the supplied bitstream is decoded once and subsequently bandwidth-limited and re-encoded prior to recording, suffers from deteriorated picture quality. If, in recording image-compressed digital signals, the transmission rate of input digital signals is less than the recording rate for the recording and/or reproducing apparatus, the method of directly recording the supplied bitstream without decoding or re-encoding suffers from deterioration in the picture quality only to the least extent. However, if the transmission rate of the input digital signals exceeds the recording rate of the recording and/or reproducing apparatus, it is indeed necessary to re-encode the bitstream and to record the re-encoded bitstream, so that, after decoding in the recording and/or reproducing apparatus, the transmission rate will be not higher than the upper limit of the disc recording rate.

If the bitstream is transmitted in a variable rate system in which the bit rate of the input digital signal is increased or decreased with time, the capacity of the recording medium can be exploited less wastefully with a disc recording apparatus

adapted for transiently storing data in a buffer and for recording the data in a burst fashion than with a tape recording system having a fixed recording rate imposed by the fixed rpm of the rotary head.

Thus, it may be predicted that, in the near future when the digital broadcast is to become the mainstream, an increasing demand will be raised for a recording and/or reproducing apparatus in which broadcast signals are recorded as digital signals, without decoding or re-encoding, as in a DataStreamer, and in which a disc is used as a recording medium.

If a recording medium having plural data, such as data of a program comprised of image data and audio data, recorded thereon, is to be reproduced by the above-described apparatus, there is presented a problem that, while the processing of determining a readout position of the AV stream from the recording medium or that of decoding the stream needs to be performed promptly responsive to a command from the user for random accessing or stream decoding, increasing difficulties are met in prompt execution of such processing with increase in the volume of data recorded on a recording medium.

#### Disclosure of the Invention

It is an object of the present invention to overcome the inconvenience in the prior art and to provide such a configuration in which the information in an I-picture in an AV stream, such as the address information, encoding information, transition

point information or the marks, is recorded as a file to enable the readout positions in the AV stream to be determined and decoded promptly as well as to enable a specified mark to be retrieved promptly.

An information processing apparatus according to the present invention includes means for generating the start address information of a domain in which the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, and means for recording the information generated by the generating means as the Clip information on the recording medium.

The start address information of the domain in which the encoding information in the AV stream is continuous is a start address of the STC sequence or the program sequence, the information correlating the time information and the address information is EP\_map or TU\_map and the time information of the characteristic picture is ClipMark.

The recording means further may record the information pertinent to an average value of the recording rate of the AV stream on the recording medium.

The information pertinent to an average value may be TS\_average\_rate.

The AV stream may be a transport stream.

The start address information of the domain in which the encoding information in the AV stream is continuous may include a start address of an STC sequence which



information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, and a step of recording the information generated by the generating step as the Clip information on the recording medium.

A program for a recording medium according to the present invention includes a step of generating the start address information of a domain in which the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, and a step of recording the information generated by the generating step as the Clip information on the recording medium.

A program according to the present invention executes a step of generating the start address information of a domain in which the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, and a step of recording the information generated by the generating step as Clip information on the recording medium.

An information processing apparatus according to the present invention includes means for reproducing the start address information of a domain in which the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, as the Clip information, and

means for controlling the outputting of the AV stream based on the Clip information reproduced by the reproducing means.

An information processing method according to the present invention includes a step of reproducing the start address information of a domain in which the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, and a step of controlling the outputting of the AV stream based on the Clip information reproduced by the reproducing means.

A program according to the present invention includes a step of reproducing the start address information of a domain in which the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, and a step of controlling the outputting of the AV stream based on the Clip information reproduced by the reproducing means.

A program according to the present invention executes a step of reproducing the start address information of a domain in which the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, and a step of controlling the outputting of the AV stream based on Clip information reproduced by the reproducing means.

The recording medium according to the present invention has recorded thereon

the start address information for a domain where the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, as the Clip information.

According to the present invention, the start address information for a domain where the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, are recorded.

Moreover, according to the present invention, the start address information for a domain where the encoding information in the AV stream is continuous, the information correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, are reproduced.

#### Brief Description of the Drawings

Fig.1 shows a configuration of an embodiment of a recording and/or reproducing apparatus according to the present invention.

Fig.2 illustrates the data format of data recorded on a recording medium by a recording and/or reproducing apparatus 1.

Fig.3 illustrates Real PlayList and Virtual PlayList.

Figs.4A, 4B and 4C illustrate the creation of the Real PlayList.

Figs.5A, 5B and 5C illustrate deletion of the Real PlayList.

Figs.6A and 6B illustrate assemble editing.

Fig.7 illustrates provision of a sub path in the Virtual PlayList.

Fig.8 illustrates the changing of the playback sequence of the PlayList.

Fig.9 illustrates a mark on the PlayList and a mark on the Clip.

Fig.10 illustrates a menu thumbnail.

Fig.11 illustrates mark added to the PlayList.

Fig.12 illustrates a mark added to the Clip.

Fig.13 illustrates the relation between the PlayList, Clip and the thumbnail file.

Fig.14 illustrates a directory structure.

Fig.15 illustrates a syntax of infr.dvr.

Fig.16 shows a syntax of DVRVolume.

Fig.17 shows a syntax of ResumeVolume.

Fig.18 shows a syntax of UIAppInfoVolume.

Fig.19 shows a table of character set values.

Fig.20 shows a syntax of TableOfPlayList.

Fig.21 shows another syntax of TableOfPlayList.

Fig.22 shows a syntax of the MakersPrivateData.

Fig.23 shows a syntax of xxxx.rpls and yyyy.vpls.

Figs.24A to 24C illustrate the PlayList.

Fig.25 shows a syntax of PlayList.

Fig.26 shows a table of `PlayList_type`.

Fig.27 shows a syntax of `UIAppInfoPlayList`.

Figs.28A to 28C illustrate flags in the `UIAppInfoPlayList` syntax shown in Fig.27.

Fig.29 illustrates a `PlayItem`.

Fig.30 illustrates a `PlayItem`.

Fig.31 illustrates a `PlayItem`.

Fig.32 shows a syntax of the `PlayItem`.

Fig.33 illustrates IN-time.

Fig.34 illustrates OUT-time.

Fig.35 shows a table of `Connection_Condition`.

Figs.36A to 36D illustrate `Connection_Condition`.

Fig.37 illustrates `BridgeSequenceInfo`.

Fig.38 shows a syntax of `BridgeSequenceInfo`.

Fig.39 illustrates `SubPlayItem`.

Fig.40 shows a syntax of `SubPlayItem`.

Fig.41 shows a table of `Mark_type`.

Fig.42 shows a syntax of `PlayListMark`.

Fig.43 shows a table of `Mark_type`.

Fig.44 illustrates `Mark_time_stamp`.

Fig.45 shows a syntax of `zzzzz.clip`.

Fig.46 shows a syntax of ClipInfo.

Fig.47 shows a table of Clip\_stream\_type.

Fig.48 illustrates offset\_SPN.

Fig.49 illustrates offset\_SPN.

Figs.50A, 50B illustrate the STC domain.

Fig.51 illustrates STC\_Info.

Fig.52 shows a syntax of STC\_Info.

Fig.53 illustrates ProgramInfo.

Fig.54 shows a syntax of ProgramInfo.

Fig.55 shows a syntax of VideoCondngInfo.

Fig.56 shows a table of Video\_format.

Fig.57 shows a table of frame\_rate.

Fig.58 shows a table of display\_aspect\_ratio.

Fig.59 shows a syntax of AudioCondngInfo.

Fig.60 shows a table of audio\_coding.

Fig.61 shows a table of audio\_component\_type.

Fig.62 shows a table of sampling\_frequency.

Fig.63 illustrates CPI.

Fig.64 illustrates CPI.

Fig.65 shows a syntax of CPI.

Fig.66 shows a table of CPI\_type.

Fig.84 shows another example of the syntax of `mark_entry()` and `representative_picture_entry()`.

Fig.85 shows the relation between ClipMark and EP\_map.

Fig.86 shows a syntax of menu.thmb and mark.thmb.

Fig.87 shows a syntax of Thumbnail.

Fig.88 shows a table of thumbnail\_picture\_format.

Figs.89A and 89B illustrate tn\_block.

Fig.90 illustrates the structure of a transport stream of DVMPEG2.

Fig.91 shows a recorder model of the transport stream of DVMPEG2.

Fig.92 shows a player model of the transport stream of DVMPEG2.

Fig.93 shows a syntax of source packet.

Fig.94 shows a syntax of TP\_extra\_header.

Fig.95 shows a table of copy permission indicator.

Fig.96 illustrates seamless connection.

Fig.97 illustrates seamless connection.

Fig.98 illustrates seamless connection.

Fig.99 illustrates seamless connection.

Fig.100 illustrates seamless connection.

Fig.101 illustrates audio overlap.

Fig.102 illustrates seamless connection employing BridgeSequence.

Fig.103 illustrates seamless connection not employing BridgeSequence.

Fig.104 shows a DVRSTD model.

Fig.105 shows a timing chart for decoding and display.

Fig.115 shows an example of ClipMark.

Fig.124 shows an application format.

Fig.125 illustrates a mark on PlayList and a mark on Clip.

Fig.126 shows another example of the ClipMark syntax.

Fig.127 shows still another example of the ClipMark syntax.

Fig.128 shows another example of the ClipInfo() syntax.

Fig.129 shows another example of the ProgramInfo().

Fig.130 shows an example of the StreamCodingInfo().

Fig.131 illustrates stream\_coding\_type.

Fig.132 illustrates the relation between EP-fine and EP-coarse.

Fig.133 illustrates the format of EP-fine and EP-coarse.

Fig.134 illustrates the format of RSPN\_EP\_fine and RSPN\_EP\_coarse.

Fig.135 illustrates the EP-coarse entry and the EP-fine entry.

Fig.136 shows another example of the EP\_Map syntax.

Fig.137 illustrates EP\_stream\_typevalues.

Fig.138 shows the syntax of EP\_map\_for\_one\_stream\_PID of EP\_map of Fig.136.

Fig.139 illustrates the meaning of the value of EP\_video\_type.

Fig.140 is a flowchart for illustrating the processing for preparing a ClipAV stream file and a ClipInformation file.

Fig.141 is a flowchart for illustrating a typical operation of preparing STC\_Info.

Fig.142 is a flowchart for illustrating a typical operation of preparing ProgramInfo.

Fig.143 is a flowchart for illustrating a typical operation of preparing EP\_map.

Fig.144 is a flowchart for illustrating the method of preparing ClipMark if, in case of encoding and recording analog AV signals, the mark\_entry()/representative\_picture\_entry() of ClipMark in Fig.75 or Fig.78 is the syntax shown in Fig.81.

Fig.145 is a flowchart for illustrating the method of preparing ClipMark if, in case of recording a transport stream input from a digital interface, the mark\_entry()/representative\_picture\_entry() of ClipMark in Fig.75 or Fig.78 is the syntax shown in Fig.81.

Fig.146 illustrates special reproduction in case of using EP\_map.

Fig.147 illustrates a player model for I-picture search employing EP\_map.

Fig.148 shows an example of a minimizing operation.

Fig.149 shows a case of erasing an unneeded stream data ahead of IN\_time in case of the minimizing operation.

Fig.150 illustrates a case of erasing an unneeded stream data at back of OUT\_time in case of the minimizing operation.

Fig.151 illustrates a medium.

### Best Mode for Carrying out the Invention

Referring to the drawings, present embodiment of the present invention will be explained in detail. Fig.1 shows a typical inner structure of a recording and/or

The multiplexed stream is e.g., an MPEG-2 transport stream or an MPEG2

If the input transport stream is re-encoded and subsequently recorded, the

transport stream, input to the terminal 13, is fed to a demultiplexer 26, which demultiplexes the input transport stream to extract a video stream (V), an audio stream (A) and the system information (S).

Of the stream (information), as extracted by the demultiplexer 26, the video stream is output to an audio decoder 27, whilst the audio stream and the system information are output to the multiplexer 16. The audio decoder 27 decodes the input transport stream to output the encoded video stream (V) to the multiplexer 16.

The audio stream and the system information, output from the demultiplexer 26 and input to the multiplexer 16, and the video stream, output by the AV encoder 15, are multiplexed, based on the input system information, and output to the multiplexed stream analysis unit 18 and to the source packetizer 19 through switch 17, as a multiplexed stream. The ensuing processing of recording an AV stream on a recording medium is the same as that of encoding and recording analog input audio and video signals, as described above, and hence is not explained here for simplicity.

The recording and/or reproducing apparatus 1 of the present embodiment records a file of the AV stream on the recording medium 100, while also recording the application database information which accounts for the file. The input information to the controller 23 is the feature information for the moving picture from the analysis unit 14, the feature information of the AV stream from the multiplexed stream analysis unit 18 and the user command information input at a terminal 24.

The feature information of the moving picture, supplied from the analysis unit

14, is generated by the analysis unit 14 when the AV encoder 15 encodes video signals. The analysis unit 14 analyzes the contents of the input video and audio signals to generate the information pertinent to the pictures characteristic of the input moving picture signals (clip mark). This information is the information indicating a picture of characteristic clip mark points, such as program start points, scene change points, CM commercial start and end points, title or telop in input video signals, and also includes a thumbnail of the picture and the information pertinent to stereo/monaural switching points and muted portions of audio signals.

The above picture indicating information is fed through controller 23 to the multiplexer 16. When multiplexing a encoded picture specified as clip mark by the controller 23, the multiplexer 16 returns the information for specifying the encoded picture on the AV stream to the controller 23. Specifically, this information is the PTS (presentation time stamp) of a picture or the address information on the AV stream of an encoded version of the picture. The controller 23 stores the sort of feature pictures and the information for specifying the encoded picture on the AV stream in association with each other.

The feature information of the AV stream from the multiplexed stream analysis unit 18 is the information pertinent to the encoding information of the AV stream to be recorded, and is recorded by an analysis unit 18. For example, the feature information includes the time stamp and address information of the I-picture in the AV stream, discontinuous point information of system time clocks, encoding parameters

of the AV stream and change point information of the encoding parameters in the AV stream. When transparently recording the transport stream, input from the terminal 13, the multiplexed stream analysis unit 18 detects the picture of the aforementioned clip mark, from the input transport stream, and generates the information for specifying a picture designated by the clip mark and its type.

The user designation information from the terminal 24 is the information specifying the playback domain, designated by the user, character letters for explaining the contents of the playback domain, or the information such as bookmarks or resuming points set by the user for his or her favorite scene.

Based on the aforementioned input information, the controller 23 creates a database of the AV stream (Clip), a database of a group (PlayList) of playback domains (PlayItem) of the AV stream, management information of the recorded contents of the recording medium 100 (info.dvr) and the information on thumbnail pictures. Similarly to the AV stream, the application database information, constructed from the above information, is processed in the ECC unit 20 and the modulation unit 21 and input to the write unit 22, which then records a database file on the recording medium 100.

The above-described application database information will be explained subsequently in detail.

When the AV stream file recorded on the recording medium 100 (files of picture data and speech data) and the application database information, thus recorded

on the recording medium 100, are reproduced by a reproducing unit 3, the controller 23 first commands a readout unit 28 to read out the application database information from the recording medium 100. The readout unit 28 reads out the application database information from the recording medium 100, which then reads out the application database information from the recording medium 100 to send the application database information through demodulation and error correction processing by a demodulating unit 29 and an ECC decoder 30 to the controller 23.

Based on the application database information, the controller 23 outputs a list of PlayList recorded on the recording medium 100 to a user interface of the terminal 24. The user selects the PlayList, desired to be reproduced, from the list of PlayLists. The information pertinent to PlayList, specified to be reproduced, is input to the controller 23. The controller 23 commands the readout unit 28 to read out the AV stream file necessary in reproducing the PlayList. In accordance with the command, the readout unit 28 reads out the corresponding AV stream from the recording medium 100 to output the read-out AV stream to the demodulating unit 29. The AV stream, thus input to the demodulating unit 29, is demodulated by preset processing and output through the processing by the ECC decoder 30 to a source depacketizer 31.

The source depacketizer 31 converts the AV stream of the application format, read out from the recording medium 100 and processed in a preset fashion, into a stream processable by the demultiplexer 26. The demultiplexer 26 outputs the system information (S), such as the video stream (V), audio stream (A) or the AV

synchronization, forming the playback domain (PlayItem) of the AV stream specified by the controller 23, to the audio decoder 27, which AV decoder 27 decodes the video stream and the audio stream to output the playback video signal and the playback audio signal to associated terminals 32, 33, respectively.

If fed from the terminal 24, as a user interface, with the information instructing random access playback or special playback, the controller 23 determines the readout position of the AV stream from the recording medium 100, based on the contents of the database (Clip) of the AV stream, to command the readout unit 28 to read out the AV stream. If the PlayList as selected by the user is to be reproduced as from a preset time point, the controller 23 commands the readout unit 28 to read out data from an I-picture having a time stamp closest to the specified time point.

When the user has selected a certain clip mark from indexing points or scene change points for the program stored in the ClipMark in the Clip Information, as when the user selects a certain picture from a list of thumbnail pictures, as demonstrated on a user interface, of the indexing points or scene change points stored in the ClipMark, the controller 23 determines the AV stream readout position from the recording medium 100 to command the readout unit 28 to read out the AV stream. That is, the controller 23 commands the readout unit 28 to read out data from an I-picture having an address closest to the address on the AV stream which has stored the picture selected by the user. The readout unit 28 reads out data from the specified address. The read-out data is processed by the demodulating unit 29, ECC decoder 30 and by

the source packetizer 19 so as to be supplied to the demultiplexer 26 and decoded by the audio decoder 27 to reproduce AV data indicated by an address of the mark point picture.

If the user has commanded fast forward playback, the controller 23 commands the readout unit 28 to sequentially read out I-picture data in the AV stream in succession based on the database (Clip) of the AV stream.

The readout unit 28 reads out data of the AV stream from a specified random access point. The so read-out data is reproduced through processing by various components on the downstream side.

The case in which the user edits the AV stream recorded on the recording medium 100 is now explained. If desired to specify a playback domain for the AV stream recorded on the recording medium 100, for example, if desired to create a playback route of reproducing a portion sung by a singer A from a song program A, and subsequently reproducing a portion sung by the same singer A from another song program B, the information pertinent to a beginning point (IN-point) and an end point (OUT-point) of the playback domain is input to the controller 23 from the terminal as a user interface. The controller 23 creates a database of the group (PlayList) of playback domains (PlayItem) of the AV streams.

When the user desires to erase a portion of the AV stream recorded on the recording medium 100, the information pertinent to the IN-point and the OUT-point of the erasure domain is input to the controller 23, which then modifies the database

of the PlayList so as to refer to only the needed AV streams. The controller 23 also commands the write unit 22 to erase an unneeded stream portion of the AV stream.

The case in which the user desires to specify playback domains of an AV stream recorded on the recording medium to create a new playback route, and to interconnect the respective playback domains in a seamless fashion, is now explained. In such case, the controller 23 creates a database of a group (PlayList) of the playback domains (PlayItem) of the AV stream and undertakes to partially re-encode and re-multiplex the video stream in the vicinity of junction points of the playback domains.

The picture information at the IN-point and that at the OUT-point of a playback domain are input from a terminal 24 to a controller 23. The controller 23 commands the readout unit 28 to read out data needed to reproduce the pictures at the IN-point and at the OUT-point. The readout unit 28 reads out data from the recording medium 100. The data so read out is output through the demodulating unit 29, ECC decoder 30 and the source packetizer 19 to the demultiplexer 26.

The controller 23 analyzes data input to the demultiplexer 26 to determine the re-encoding method for the video stream (change of picture\_coding\_type and assignment of the quantity of encoding bits for re-encoding) and the re-multiplexing system to send the system to the AV encoder 15 and to the multiplexer 16.

The demultiplexer 26 then separates the input stream into the video stream (V), audio stream (A) and the system information (S). The video stream may be classed into data input to the audio decoder 27 and data input to the multiplexer 16. The

former is data needed for re-encoding, and is decoded by the audio decoder 27, with the decoded picture being then re-encoded by the AV encoder 15 and thereby caused to become a video stream. The latter data is data copied from an original stream without re-encoding. The audio stream and the system information are directly input to the multiplexer 16.

The multiplexer 16 multiplexes an input stream, based on the information input from the controller 23, to output a multiplexed stream, which is processed by the ECC unit 20 and the modulation unit 21 so as to be sent to the write unit 22. The write unit 22 records an AV stream on the recording medium 100 based on the control signals supplied from the controller 23.

The application database information and the operation based on this information, such as playback and editing, are hereinafter explained. Fig.2 shows the structure of an application format having two layers, that is PlayList and Clip, for AV stream management. The Volume Information manages all Clips and PlayLists in the disc. Here, one AV stream and the ancillary information thereof, paired together, is deemed to be an object, and is termed Clip. The AV stream file is termed a Clip AV stream file, with the ancillary information being termed the Clip Information file.

One Clip AV stream file stores data corresponding to an MPEG-2 transport stream arranged in a structure prescribed by the application format. By and large, a file is treated as a byte string. The contents of the Clip AV stream file are expanded on the time axis, with entry points in the Clip (I-picture) being mainly specified on the

time basis. When a time stamp of an access point to a preset Clip is given, the Clip Information file is useful in finding the address information at which to start data readout in the Clip AV stream file.

Referring to Fig.3, PlayList is now explained, which is provided for a user to select a playback domain desired to be viewed from the Clip and to edit the playback domain readily. One PlayList is a set of playback domains in the Clip. One playback domain in a preset Clip is termed PlayItem and is represented by a pair of an IN-point and an OUT-point on the time axis. So, the PlayList is formed by a set of plural PlayItems.

The PlayList is classified into two types, one of which is Real PlayList and the other of which is Virtual PlayList. The Real PlayList co-owns stream portions of the Clip it is referencing. That is, the Real PlayList takes up in the disc the data capacity corresponding to a stream portion of the Clip it is referencing and, when Real PlayList is erased, the data of the stream portion of the Clip it is referencing is also erased.

The Virtual PlayList is not co-owning Clip data. Therefore, if the Virtual PlayList is changed or erased, the contents of the Clip are in no way changed.

The editing of the Real Playlist is explained. Fig.4A shows creation of Real PlayList and, if the AV stream is recorded as a new Clip, the Real PlayList which references the entire Clip is a newly created operation.

Fig.4B shows the division of the real PlayList, that is the operation of dividing the Real PlayList at a desired point to split the Real PlayList in two Real PlayLists.

This division operation is performed when two programs are managed in one clip managed by a sole PlayList and when the user intends to re-register or re-record the programs as separate individual programs. This operation does not lead to alteration of the Clip contents, that is to division of the Clip itself.

Fig.4C shows the combining operation of the Real PlayList which is the operation of combining two Real PlayLists into one new Real PlayList. This combining operation is performed such as when the user desires to re-register two programs as a sole program. This operation does not lead to alteration of the Clip contents, that is to combining the clip itself into one.

Fig.5A shows deletion of the entire Real PlayList. If the operation of erasing the entire preset Real PlayList, the associated stream portion of the Clip referenced by the deleted Real PlayList is also deleted.

Fig.5B shows partial deletion of the Real PlayList. If a desired portion of the Real PlayList is deleted, the associated PlayItem is altered to reference only the needed Clip stream portion. The corresponding stream portion of the Clip is deleted.

Fig.5C shows the minimizing of the Real PlayList. It is an operation of causing the PlayItem associated with the Real PlayList to reference only the stream portion of the Clip needed for Virtual PlayList. The corresponding stream portion of the Clip not needed for the Virtual PlayList is deleted.

If the Real PlayList is changed by the above-described operation such that the stream portion of the Clip referenced by the Real PlayList is deleted, there is a

If there exist two Real PlayLists 1, 2 and clips 1, 2 associated with the respective Real PlayLists, the user specifies a preset domain in the Real PlayList 1 (domain from IN1 to OUT1: PlayItem 1) as the playback domain, and also specifies, as the domain to be reproduced next, a preset domain in the Real PlayList 2 (domain from IN2 to OUT2: PlayItem 2) as the playback domain, as shown in Fig.6A, a sole Virtual PlayList made up of PlayItem 1 and the PlayItem2 is prepared, as shown in

Fig.6B.

The re-editing of the Virtual PlayList is now explained. The re-editing may be enumerated by alteration of IN- or OUT points in the Virtual PlayList, insertion or appendage of new PlayItems to the Virtual PlayList and deletion of PlayItems in the Virtual PlayList. The Virtual PlayList itself may also be deleted.

Fig.7 shows the audio dubbing (post recording) to the Virtual PlayList. It is an operation of registering the audio post recording to the Virtual PlayList as a sub path. This audio post recording is supported by the application software. An additional audio stream is added as a sub path to the AV stream of the main path of the Virtual PlayList.

Common to the Real PlayList and the Virtual PlayList is an operation of changing (moving) the playback sequence of the PlayList shown in Fig.8. This operation is an alteration of the playback sequence of the PlayList in the disc (volume) and is supported by TableOfPlayList as defined in the application format, as will be explained subsequently with reference to e.g., Fig.20. This operation does not lead to alteration of the Clip contents.

The mark (Mark) is now explained. The mark is provided for specifying a highlight or characteristic time in the Clip and in the PlayList, as shown in Fig.9. The mark added to the Clip is termed the ClipMark. The ClipMark is e.g., a program indexing point or a scene change point for specifying a characteristic scene ascribable to contents in the AV stream. The ClipMark is generated by e.g., the analysis unit 14

of Fig.1. When the PlayList is reproduced, the mark of the Clip referenced by the PlayList may be referenced and used.

The mark appended to the PlayList is termed the PlayListMark (play list mark). The PlayListMark is e.g., a bookmark point or a resuming point as set by the user. The setting of the mark to the Clip and to the PlayList is by adding a time stamp indicating the mark time point to the mark list. On the other hand, mark deletion is removing the time stamp of the mark from the mark list. Consequently, the AV stream is in no way changed by mark setting or by mark deletion.

As another format of the ClipMark, a picture referenced by the ClipMark may be specified on the address basis in the AV stream. Mark setting on the Clip is by adding the address basis information indicating the picture of the mark point to the mark list. On the other hand, mark deletion is removing the address basis information indicating the mark point picture from the mark list. Consequently, the AV stream is in no way changed by mark setting or by mark deletion.

A thumbnail is now explained. The thumbnail is a still picture added to the Volume, PlayList and Clip. There are two sorts of the thumbnail, one of them being a thumbnail as a representative picture indicating the contents. This is mainly used in a main picture in order for the user to select what he or she desired to view on acting on a cursor, not shown. Another sort of the thumbnail is a picture indicating a scene pointed by the mark.

The Volume and the respective PlayLists need to own representative pictures.

As the representative picture of the PlayList, it may be contemplated to use the initial picture of the PlayList as the thumbnail (representative picture). However, the leading picture at the playback time of 0 is not necessarily an optimum picture representing the contents. So, the user is allowed to set an optional picture as a thumbnail of the PlayList. Two sorts of the thumbnails, that is the thumbnail as a representative picture indicating the Volume and the thumbnail as a representative picture indicating PlayList, are termed menu thumbnails. Since the menu thumbnails are demonstrated frequently, these thumbnails need to be read out at an elevated speed from the disc. Thus, it is efficient to store the totality of the menu thumbnails in a sole file. It is unnecessary for the menu thumbnails to be pictures extracted from the moving pictures in the volume, but may be a picture captured from a personal computer or a digital still camera, as shown in Fig.10.

On the other hand, the Clip and the PlayList need be marked with plural marks, whilst the pictures of the mark points need to be readily viewed in order to grasp the contents of the mark positions. The picture indicating such mark point is termed a

mark thumbnail. Therefore, the picture which is the original of the mark thumbnail is mainly an extracted mark point picture rather than a picture captured from outside.

Fig.11 shows the relation between the mark affixed to the PlayList and the mark thumbnail, whilst Fig.12 shows the relation between the mark affixed to the Clip and the mark thumbnail. In distinction from the menu thumbnail, the mark thumbnail is used in e.g., a sub-menu for representing details of the PlayList, while it is not requested to be read out in a short access time. So, whenever a thumbnail is required, the recording and/or reproducing apparatus 1 opens a file and reads out a portion of the file, while there is no problem presented even if file opening and reading out a portion of the file by the recording and/or reproducing apparatus 1 takes some time.

For decreasing the number of files present in a volume, it is preferred for the totality of the mark thumbnails to be stored in one file. While the PlayList may own one menu thumbnail and plural mark thumbnails, the user is not required to select the Clip directly (usually, the Clip is selected through PlayList), and hence there is no necessity of providing menu thumbnails.

Fig.13 shows the relation between the menu thumbnails, mark thumbnails, PlayList and Clips. In the menu thumbnail file are filed menu thumbnails provided from one PlayList to another. In the menu thumbnail file is contained a volume thumbnail representing the contents of data recorded on the disc. In the menu thumbnail file are filed thumbnails created from one PlayList to another and from one Clip to another.

The TU\_map has a list of time unit (TU) data which is derived from the arrival time point of the transport packet input through a digital interface. This affords the relation between the arrival-time-based time and the data address in the AV stream. When the recording and/or reproducing apparatus 1 records an input AV stream, and the syntax of the stream cannot be analyzed, a TU\_map is created and recorded on the disc.

It is useful for a reproducing apparatus (recording and/or reproducing apparatus 1 of Fig.1) to know the contents of an AV stream prior to its decoding. These contents include e.g., values of the PID of a transport packet transmitting an audio or video elementary stream or the type of the video or audio components, such as HDTV video or MPEG-2 AAC audio stream. This information is useful for creating a menu screen for illustrating to the user the contents of the PlayList referencing the AV stream. It is similarly useful for setting the initial state of the AV decoder and the demultiplexer of the respective apparatus.

For this reason, the Clip Information file owns ProgramInfo for illustrating the program contents.

It may be an occurrence that program contents be changed in the AV stream file in which the MPEG-2 transport stream is stored. For example, the PID of the transport packet transmitting the video elementary stream may be changed, or the component type of the video stream may be changed from SDTV to HDTV.

The ProgramInfo stores the information on change points of program contents in the AV stream file. The domain of the AV stream file in which the program contents remain constant is termed program-sequence.

This program-sequence is used in an AV stream file having EP\_map and is optional in an AV stream file having TU\_map.

The present embodiment defines the self-encoding stream format (SESF). This SESF is used for encoding analog input signals and for decoding a digital input signals for subsequently encoding the decoded signal into an MPEG-2 transport stream.

The SESF defines an elementary stream pertinent to the MPEG-2 transport stream and the AV stream. When the recording and/or reproducing apparatus 1 encodes and records the SESF stream, an EP\_map is created and recorded on the disc.

A digital broadcast stream uses one of the following systems for recording on the recording medium 100: First, the digital broadcast stream is transcoded into an SESF stream. In this case, the recorded stream must conform to SESF and EP\_map must be prepared and recorded on the disc.

Alternatively, an elementary stream forming a digital broadcast stream is transcoded to a new elementary stream and re-multiplexed to a new transport stream conforming to the stream format prescribed by the organization for standardizing the digital broadcast stream. In this case, an EP\_map must be created and recorded on the disc.

For example, it is assumed that the input stream is an MPEG-2 transport stream conforming to the ISDB (standard appellation of digital BS of Japan), with the transport stream containing the HDTV video stream and the MPEG AAC audio stream. The HDTV video stream is transcoded to an SDTV video stream, which SDTV video stream and the original AAC audio stream are re-multiplexed to TS. The SDTV stream and the transport stream both need to conform to the ISDB format.

Another system of recording the digital broadcast stream on the recording medium 100 is to make transparent recording of the input transport stream, that is to record the input transport stream unchanged, in which case the EP\_map is formulated and recorded on the disc.

Alternatively, the input transport stream is recorded transparently, that is an input transport stream is recorded unchanged, in which case TU\_map is created and recorded on the disc.

The directory and the file are hereinafter explained. The recording and/or reproducing apparatus 1 is hereinafter described as DVR (digital video recording). Fig.14 shows a typical directory structure on the disc. The directories of the disc of

the DVR may be enumerated by a root directory including "DVR" directory, and the "DVR" directory, including "PLAYLIST" directory, "CLIPINF" directory, "M2TS" directory and "DATA" directory, as shown in Fig.14. Although directories other than these may be created below the root directory, these are discounted in the application format of the present embodiment.

Below the "DATA" directory, there are stored all files and directories prescribed by the DVR application format. The "DVR" directory includes four directories. Below the "PLAYLIST" directory are placed Real PlayList and Virtual PlayList database files. The latter directory may exist in a state devoid of PlayList.

Below "CLIPINF" is placed a Clip database. This directory, too, may exist in a state devoid of AV stream files. In the "DATA" directory, there are stored files of data broadcast, such as digital TV broadcast.

The "DVR" directory stores the following files: That is, an "info.dvr" is created below the DVR directory to store the comprehensive information of an application layer. Below the DVR directory, there must be a sole info.dvr. The filename is assumed to be fixed to info.dvr. The "menu.thmb" stores the information pertinent to the menu thumbnails. Below the DVR directory, there must be 0 or 1 mark thumbnail. The filename is assumed to be fixed to "menu.thmb". If there is no menu thumbnail, this file may not exist.

The "mark.thmb" file stores the information pertinent to the mark thumbnail picture. Below the DVR directory, there must be 0 or 1 mark thumbnail. The

filename is assumed to be fixed to "menu.thmb". If there is no menu thumbnail, this file may not exist.

The "PLAYLIST" directory stores two sorts of the PlayList files which are Real PlayList and Virtual PlayList. An "xxxxx.rpls" file stores the information pertinent to one Real PlayList. One file is created for each Real PlayList. The filename is "xxxxx.rpls", where "xxxxx" denotes five numerical figures from 0 to 9. A file extender must be "rpls".

The "yyyyy.vpls" stores the information pertinent to one Virtual PlayList. One file with a filename "yyyyy.vpls" is created from one Virtual PlayList to another, where "yyyyy" denotes five numerical figures from 0 to 9. A file extender must be "vpls".

The "CLIPINF" directory stores one file in association with each AV stream file. The "zzzzz.clpi" is a Clip Information file corresponding to one AV stream file (Clip AV stream file or Bridge-Clip stream file). The filename is "zzzzz.clpi", where "zzzzz" denotes five numerical figures from 0 to 9. A file extender must be "clpi".

The "M2TS" directory stores an AV stream file. The "zzzzz.m2ts" file is an AV stream file handled by the DVR system. This is a Clip AV stream file or a Bridge-Clip AV stream file. The filename is "zzzzz.m2ts", where "zzzzz" denotes five numerical figures from 0 to 9. A file extender must be "m2ts".

The "DATA" directory stores data transmitted from data broadcasting. This data may, for example, be XML or MPEG files.

The syntax and the semantics of each directory (file) are now explained. Fig.15

shows the syntax of the "info.dvr" file. The "info.dvr" file is made up of three objects, that is DVRVolume(), TableOfPlayLists() and MakersPrivateData().

The syntax of info.dvr shown in Fig.15 is explained. The TableOfPlayLists\_Start\_address indicates the leading address of the TableOfPlayLists() in terms of the relative number of bytes from the leading byte of the "info.dvr" file. The relative number of bytes is counted beginning from 0.

The MakersPrivateData\_Start\_address indicates the leading address of the MakersPrivateData(), in terms of the relative number of bytes as from the leading byte of the "info.dvr" file. The relative number of bytes is counted from 0. The padding\_word is inserted in association with the syntax of "info.dvr". N1 and N2 are optional positive integers. Each padding word may assume an optional value.

The DVRVolume() stores the information stating the contents of the volume (disc). Fig.16 shows the syntax of the DVRVolume. The syntax of the DVRVolume(), shown in Fig.16, is now explained. The version\_number indicates four character letters indicating the version numbers of the DVRVolume(). The version\_number is encoded to "0045" in association with ISO646.

Length is denoted by 32-bit unsigned integers indicating the number of bytes from directly after the length field to the trailing end of DVRVolume().

The ResumeVolume() memorizes the filename of the Real PlayList or the Virtual PlayList reproduced last in the Volume. However, the playback position when the user has interrupted playback of the Real PlayList or the Virtual PlayList is stored

in the resume-mark defined in the `PlayListMark()` (see Figs.42 and 43).

Fig.17 shows the syntax of the `ResumeVolume()`. The syntax of the `ResumeVolume()` shown in Fig.17 is explained. The `valid_flag` indicates that the `resume_PlayList_name` field is valid or invalid when this 1-bit flag is set to 1 or 0, respectively.

The 10-byte field of `resume_PlayList_name` indicates the filename of the Real PlayList or the Virtual PlayList to be resumed.

The `UIAppInfoVolume` in the syntax of the `DVRVolume()`, shown in Fig.16, stores parameters of the user interface application concerning the Volume. Fig.18 shows the syntax of the `UIAppInfoVolume`, the semantics of which are now explained. The 8-bit field of `character_set` indicates the encoding method for character letters encoded in the `Volume_name` field. The encoding method corresponds to the values shown in Fig.19.

The 8-bit field of the `name_length` indicates the byte length of the Volume name indicated in the `Volume_name` field. The `Volume_name` field indicates the appellation of the Volume. The number of bytes of the number of the `name_length` counted from left of the field is the number of valid characters and indicates the volume appellation. The values next following these valid character letters may be any values.

The `Volume_protect_flag` is a flag indicating whether or not the contents in the Volume can be shown to the user without limitations. If this flag is set to 1, the contents of the Volume are allowed to be presented (reproduced) to the user only in

case the user has succeeded in correctly inputting the PIN number (password). If this flag is set to 0, the contents of the Volume are allowed to be presented to the user even in case the PIN number is not input by the user.

If, when the user has inserted a disc into a player, this flag has been set to 0, or the flag is set to 1 but the user has succeeded in correctly inputting the PIN number, the recording and/or reproducing apparatus 1 demonstrates a list of the PlayList in the disc. The limitations on reproduction of the respective PlayLists are irrelevant to the Volume\_protect\_flag and are indicated by playback\_control\_flag defined in the UIAppInfoVolume.

The PIN is made up of four numerical figures of from 0 to 9, each of which is coded in accordance with ISO/IEC 646. The ref\_thumbnail\_index field indicates the information of a thumbnail picture added to the Volume. If the ref\_thumbnail\_index field is of a value other than 0xFFFF, a thumbnail picture is added to the Volume. The thumbnail picture is stored in a menu.thumb file. The picture is referenced using the value of the ref\_thumbnail\_index in the menu.thumb file. If the ref\_thumbnail\_index field is 0xFFFF, it indicates that a thumbnail picture has been added to the Volume.

The TableOfPlayList() in the info.dvr syntax shown in Fig.15 is explained. The TableOfPlayList() stores the filename of the PlayList (Real PlayList and Virtual PlayList). All PlayList files recorded in the Volume are contained in the TableOfPlayList(), which TableOfPlayList() indicates the playback sequence of the default of the PlayList in the Volume.

Fig.20 shows the syntax of the TableOfPlayList(), which is now explained. The version\_number of the TableOfPlayList() indicates four character letters indicating the version numbers of the TableOfPlayLists. The version\_number must be encoded to "0045" in accordance with ISO 646.

Length is a unsigned 32-bit integer indicating the number of bytes of the TableOfPlayList() from directly after the length field to the trailing end of the TableOfPlayList(). The 16-bit field of the number\_of\_PlayLists indicates the number of loops of the for-loop inclusive of the PlayList\_file\_name. This numerical figure must be equal to the number of PlayLists recorded in the Volume. The 10-byte numerical figure of the PlayList\_file\_name indicates the filename of the PlayLists.

Fig.21 shows another configuration of the syntax of the TableOfPlayList(). The syntax shown in Fig.21 is comprised of the syntax shown in Fig.20 in which is contained the UIAppInfoPlayList. By such structure including the UIAppInfoPlayList, it becomes possible to create a menu picture simply on reading out the TableOfPlayLists. The following explanation is premised on the use of the syntax shown in Fig.20.

The MakersPrivateData in the info.dvr shown in Fig.15 is explained. The MakersPrivateData is provided to permit the maker of the recording and/or reproducing apparatus 1 to insert private data of the maker in the MakersPrivateData() for special applications of different companies. The private data of each maker has standardized maker\_ID for identifying the maker who has defined it. The



The real PlayList file and the Virtual PlayList file, in other words, xxxxx.rpls and yyyyy.vpls, are explained. Fig.23 shows the syntax of xxxxx.rpls (Real PlayList) and yyyyy.vpls (Virtual PlayList), which are of the same syntax structure. Each of the xxxxx.rpls and yyyyy.vpls is made up of three objects, that is PlayList(), PlayListMark() and MakersPrivateData().

The `PlayListMark_Start_address` indicates the leading address of the `PlayListMark()`, in terms of the relative number of bytes from the leading end of the `PlayList` file as a unit. The relative number of bytes is counted from zero.

The `MakersPrivateData_Start_address` indicates the leading address of the `MakersPrivateData()`, in terms of the relative number of bytes from the leading end of the `PlayList` file as a unit. The relative number of bytes is counted from zero.

The `padding_word` (padding word) is inserted in accordance with the syntax of the `PlayList` file, with `N1` and `N2` being optional positive integers. Each padding word may assume an optional value.

`PlayList` will be further explained in the following although it has been explained briefly. A playback domain in all Clips except Bridge-Clip must be referred by all `PlayLists` in the disc. Also, two or more Real `PlayLists` must not overlap the playback domains shown by their `PlayItems` in the same Clip.

Reference is made to Figs.24A, 24B and 24C. For all Clips, there exist corresponding Real `PlayLists`, as shown in Fig.24A. This rule is observed even after the editing operation has come to a close, as shown in Fig.24B. Therefore, all Clips must be viewed by referencing one of Real `PlayLists`.

Referring to Fig.24C, the playback domain of the Virtual `PlayList` must be contained in the playback domain and in the Bridge-Clip playback domain. There must not be present in the disc Bridge-Clip not referenced by any Virtual `PlayList`.

The Real `PlayList`, containing the list of the `PlayItem`, must not contain

The `PlayItem_id` corresponding to the preset `PlayItem()` is defined by the sequence in which the `PlayItem()` appears in the for-loop containing the `PlayItem()`. The `PlayItem_id` begins with 0. The `number_of_SubPlayItems` is a 16-bit field

indicating the number of SubPlayItem in the PlayList. This value is 0 or 1. An additional audio stream path (audio stream path) is a sort of a sub path.

The UIAppInfoPlayList of the PlayList syntax shown in Fig.25 is explained. The UIAppInfoPlayList stores parameters of the user interface application concerning the PlayList. Fig.27 shows the syntax of the UIAppInfoPlayList, which is now explained. The character\_set is an 8-bit field indicating the method for encoding character letters encoded in the PlayList\_name field. The encoding method corresponds to the values conforming to the table shown in Fig.19.

The name\_length is an 8-bit field indicating the byte length of the PlayList name indicated in the PlayList\_name field. The PlayList\_name field shows the appellation of the PlayList. The number of bytes of the number of the name\_length counted from left of the field is the number of valid characters and indicates the PlayList appellation. The values next following these valid character letters may be any values.

The record\_time\_and\_date is a 56-bit field storing the date and time on which the PlayList was recorded. This field is 14 numerical figures for year/ month/ day/ hour/minute/ second encoded in binary coded decimal (BCD). For example, 2001/ 12/ 23:01:02:03 is encoded to "0x20011223010203".

The duration is a 24-bit field indicating the total replay time of the PlayList in terms of hour/ minute/ second as a unit. This field is six numerical figures encoded in binary coded decimal (BCD). For example, 01:45:30 is encoded to "0x014530".

The valid\_period is a 32-bit field indicating the valid time periods of the

PlayList. This field is 8 numerical figures encoded in 4-bit binary coded decimal (BCD). The valid\_period is used in the recording and/or reproducing apparatus 1 e.g., when the PlayList, for which the valid period has lapsed, is to be automatically erased. For example, 2001/05/07 is encoded to "0x20010507".

The maker\_ID is a 16-bit unsigned integer indicating the maker of the DVR player (recording and/or reproducing apparatus 1) which has been the last to update its PlayList. The value encoded to maker\_ID is assigned to the licensor of the DVR format. The maker\_code is a 16-bit unsigned integer indicating the model number of the DVR player which has been the last to update the PlayList. The value encoded to the maker\_code is determined by the maker who has received the license of the DVR format.

If the flag of the playback\_control\_flag is set to 1, its PlayList is reproduced only when the user successfully entered the PIN number. If this flag is set to 0, the user may view the PlayList without the necessity of inputting the PIN number.

If the write\_protect\_flag is set to 1, the contents of the PlayList are not erased nor changed except the write\_protect\_flag. If this flag is set to 0, the user is free to erase or change the PlayList. If this flag is set to 1, the recording and/or reproducing apparatus 1 demonstrates a message requesting re-confirmation by the user before the user proceeds to erase, edit or overwrite the PlayList.

The Real PlayList, in which the write\_protect\_flag is set to 0, may exist, the Virtual PlayList, referencing the Clip of the Real PlayList may exist, and the

write\_protect\_flag of the Virtual PlayList may be set to 1. If the user is desirous to erase the Real PlayList, the recording and/or reproducing apparatus 1 issues an alarm to the user as to the presence of the aforementioned Virtual PlayList or "minimizes" the Real PlayList before erasing the Real PlayList.

If is\_played\_flag is set to 1, as shown in Fig.28B, it indicates that the PlayList was reproduced at least once since it was recorded, whereas, if it is set to 0, it indicates that the PlayList was not reproduced even once since it was recorded.

Archive is a two-bit field indicating whether the PlayList is an original or a copy, as shown in Fig.28C. The field of ref\_thumbnail\_index indicates the information of a thumbnail picture representative of the PlayList. If the ref\_thumbnail\_index field is of a value other than 0xFFFF, a thumbnail picture representative of the PlayList is added in the PlayList, with the PlayList being stored in the menu.thmb file. The picture is referenced using the value of ref\_thumbnail\_index in the menu.thmb file. If the ref\_thumbnail\_index field is 0xFFFF, no thumbnail picture representative of the PlayList is added in the PlayList.

The PlayItem is hereinafter explained. One PlayItem() basically contains the following data: Clip\_Information\_file\_name for specifying the filename of the Clip, IN-time and OUT-time, paired together to specify the playback domain of Clip, STC\_sequence\_id referenced by IN-time and OUT-time in case the CPI\_type defined in PlayList() is EP\_map type, and Connection\_Condition indicating the connection condition of previous PlayItem and current PlayItem.

Fig.30 shows such a case in which the `CPI_type` defined by `PlayList()` and, if the current `PlayItem` has the `BridgeSequence()`, the rules as now explained are applied. The `IN_time` of the `PlayItem` previous to the current `PlayItem`, shown as `IN_time1`, indicates the time in Bridge-Clip specified in the `BridgeSequenceInfo()` of the current `PlayItem`. This `OUT_time` must obey the encoding limitations which will be explained subsequently.

If the CPI\_type of PlayList() is TU\_map type, the IN\_time and OUT\_time of PlayItem, paired together, indicate the time on the same Clio AV stream, as shown in Fig.31.

The PlayItem syntax is as shown in Fig.32. As to the syntax of the PlayItem,

shown in Fig.32, the field of the Clip\_information\_file\_name indicates the filename of the Clip Information. The Clip\_stream\_type defined by the ClipInfo() of this Clip Information file must indicate the Clip AV stream.

The STC\_sequence\_id is an 8-bit field and indicates the STC\_sequence\_id of the continuous STC domain referenced by the PlayItem. If the CPI\_type specified in the PlayList() is TU\_map type, this 8-bit field has no meaning and is set to 0. IN\_time is a 32-bit field and used to store the playback start time of PlayItem. The semantics of IN\_time differs with CPI\_type defined in the PlayList(), as shown in Fig.33.

OUT\_time is a 32-bit field and is used to store the playback end time of PlayItem. The semantics of OUT\_time differs with CPI\_type defined in the PlayList(), as shown in Fig.34.

Connection\_condition is a 2-bit field indicating the connection condition between the previous PlayItem and the current PlayItem, as shown in Fig.35. Figs.36A to 36D illustrate various states of Connection\_condition shown in Fig.35.

BridgeSequenceInfo is explained with reference to Fig.37. This BridgeSequenceInfo is the ancillary information of the current PlayItem and includes the following information. That is, BridgeSequenceInfo includes Bridge\_Clip\_Information\_file\_name for specifying the Bridge\_Clip AV file and a Bridge\_Clip\_Information\_file\_name specifying the corresponding Clip Information file (Fig.45).

It is also an address of a source packet on the Clip AV stream referenced by the

previous PlayItem. Next to this source packet is connected the first source packet of the Bridge-Clip AV stream. This address is termed the RSPN\_exit\_from\_previous\_Clip. It is also an address of the source packet on the Clip AV stream referenced by the current PlayItem. Ahead of this source packet is connected the last source packet of the Bridge\_clip AV stream file. This address is termed RSPN\_enter\_to\_current\_Clip.

In Fig.37, RSPN\_arrival\_time\_discontinuity indicates an address of a source packet in the Bridge\_Clip AV stream where there is a discontinuous point in the arrival time base. This address is defined in the ClipInfo() (Fig.46).

Fig.38 shows the syntax of the BridgeSequenceInfo. Turning to the syntax of BridgeSequenceInfo shown in Fig.38, the field of Bridge\_Clip\_Information\_file\_name indicates the filename of the Clip Information file corresponding to the Bridge\_Clip\_Information\_file. The Clip\_stream\_type defined in ClipInfo() of this Clip information file must indicate 'Bridge\_Clip AV stream'.

The 32-bit field of the RSPN\_exit\_from\_previous\_Clip is a relative address of a source packet on the Clip AV stream referenced by the previous PlayItem. Next to this source packet is connected the first source packet of the Bridge\_Clip AV stream file. The RSPN\_exit\_from\_previous\_Clip has a size based on the source packet number as a unit, and is counted with the value of the offset\_SPN defined in the ClipInfo() from the first source packet of the Clip AV stream file referenced by the previous PlayItem.

The 32-bit field of `RSPN_enter_to_curent_Clip` is the relative address of the source packet on the Clip AV stream referenced by the current `PlayItem`. Ahead of this source packet is connected the last source packet of the `Bridge_Clip_AV` stream file. The `RSPN_enter_to_curent_Clip` has a size that is based on the source packet number as a unit. The `RSPN_enter_to_curent_Clip` is counted with the value of the `offset_SPN`, defined in the `ClipInfo()` from the first source packet of the Clip AV stream file referenced by the current `PlayItem`, as an initial value.

The `SubPlayItem` is explained with reference to Fig.39. The use of `SubPlayItem()` is permitted only if the `CPI_type` of the `PlayList()` is the `EP_map` type. In the present embodiment, `SubPlayItem` is used only for audio post recording. The `SubPlayItem()` includes the following data. First, it includes `Clip_Information_file_name` for specifying the Clip referenced by the sub path in the `PlayList`.

It also includes `SubPath_IN_time` and `SubPath_OUT_time` for specifying the sub path playback domain in the Clip. Additionally, it includes `sync_PlayItem_id` and `start_PTS_of_PlayItem` for specifying the time of starting the sub path reproduction on the main path time axis. The Clip AV stream, referenced by the sub path, must not contain STC discontinuous points (discontinuous points of the system time base). The clocks of audio samples of the Clip used in the sub path are locked to the clocks of the audio samples of the main path.

Fig.40 shows the syntax of the `SubPlayItem`. Turning to the syntax of the

An 8-bit field of `sync_PlayItem_id` indicates the sub path type. Here, only '0x00' is set, as shown in Fig.41, while other values are reserved for future use.

A 32-bit field of `sync_start_PTS_of_PlayItem` denotes the time of playback start of the sub path on the time axis of the main path, and denotes the upper 32 bits of the PTS (presentation time stamp) on the `PlayItem` referenced by the `sync_PlayItem_id`. The upper 32 bit field of the `SubPath_IN_time` stores the playback start time of the sub path. `SubPath_IN_time` denotes upper 32 bits of the PTS of 33 bits corresponding to the first presentation unit in the sub path.

**Presentation end TS = PTS OUT + AU duration**

where PTS\_out is the PTS of the 33 bit length corresponding to the last presentation unit of the SubPath and AU\_duration is the 90 kHz based display period of the last

presentation unit of the SubPath.

Next, `PlayListMark()` in the syntax of `xxxxx.rpls` and `yyyyy.vpls` shown in Fig.23 is explained. The mark information pertinent to the `PlayList` is stored in this `PlayListMark`. Fig.42 shows the syntax of `PlayListMark`. Turning to the syntax of the `PlayListMark` shown in Fig.42, `version_number` is four character letters indicating the version number of this `PlayListMark()`. The `version_number` must be encoded to "0045" in accordance with ISO 646.

`Length` is an unsigned 32-bit integer indicating the number of bytes of `PlayListMark()` from directly after the length field to the trailing end of the `PlayListMark()`. The `number_of_PlayListMarks` is a 16-bit unsigned integer indicating the number of marks stored in the `PlayListMark`. The `number_of_PlayListMarks` may be zero. The `mark_type` is an 8-bit field indicating the mark type and is encoded in the table shown in Fig.43.

A 32-bit field of `mark_time_stamp` stores a time stamp indicating the point specified by the mark. The semantics of the `mark_time_stamp` differ with `CPI_type` defined in the `PlayList()`, as shown in Fig.44. The `PlayItem_id` is an 8-bit field specifying the `PlayItem` where the mark is put. The values of `PlayItem_id` corresponding to a preset `PlayItem` is defined in the `PlayList()` (see Fig.25).

An 8-bit field of `character_set` shows the encoding method of character letters encoded in the `mark_name` field. The encoding method corresponds to values shown in Fig.19. The 8-bit field of `name_length` indicates the byte length of the mark name

The field of the `ref_thumbnail_index` denotes the information of the thumbnail picture added to the mark. If the field of the `ref_thumbnail_index` is not 0xFFFF, a thumbnail picture is added to its mark, with the thumbnail picture being stored in the `mark.thmb` file. This picture is referenced in the `mark.thmb` file, using the value of `ref_thumbnail_index`, as explained subsequently. If the `ref_thumbnail_index` field is 0xFFFF, it indicates that no thumbnail picture is added to the mark.

Turning to the syntax of `zzzzz.clpi` (Clip Information file) shown in Fig.45 is explained. The `ClipInfo_Start_address` indicates the leading end address of `ClipInfo()` with the relative number of bytes from the leading end byte of the `zzzzz.clpi` file as a unit. The relative number of bytes is counted from zero.

The STC Info Start address indicates the leading end address of STC Info

Turning to the syntax of the ClipInfo shown in Fig.46, version\_number is the four character letters indicating the version number of this ClipInfo(). The

version\_number must be encoded to "0045" in accordance with the ISO 646. Length is a 32-bit unsigned integer indicating the number of bytes of ClipInfo() from directly at back of the length field to the trailing end of the ClipInfo(). An 8-bit field of Clip\_stream\_type indicates the type of the AV stream corresponding to the Clip Information file, as shown in Fig.47. The stream types of the respective AV streams will be explained subsequently.

The 32-bit field of offset\_SPN gives an offset value of the source packet number of the first source packet number of the first source packet of the AV stream (Clip AV stream or the Bridge-Clip AV stream). When the AV stream file is first recorded on the disc, this offset\_SPN must be zero.

Referring to Fig.48, when the beginning portion of the AV stream file is erased by editing, offset\_SPN may assume a value other than 0. In the present embodiment, the relative source packet number (relative address) referencing the offset\_SPN is frequently described in the form of RSPNxxx, where xxx is modified such that RSPN\_xxx is RAPN\_EP\_start. The relative source packet number is sized with the source packet number as a unit and is counted from the first source packet number of the AV stream file with the value of the offset\_SPN as the initial value.

The number of source packets from the first source packet of the AV stream file to the source packet referenced by the relative source packet number (SPN\_xxx) is calculated by the following equation:

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}.$$



point when the first source packet of the AV stream file was recorded. The  $\text{size\_clip}(t)$  is  $10 \times 192$  bytes and  $\alpha$  is a constant dependent on  $\text{TS\_average\_rate}$ .

If  $\text{time\_controlled\_flag}$  is set to 0, it indicates that the recording mode is not controlling so that the time lapse of recording is proportionate to the file size of the AV stream. For example, the input transport stream is recorded in a transparent fashion.

If  $\text{time\_controlled\_flag}$  is set to 1, the 24-bit field of  $\text{TS\_average\_rate}$  indicates the value of  $\text{TS\_average\_rate}$  used in the above equation. If  $\text{time\_controlled\_flag}$  is set to 0, this field has no meaning and must be set to 0. For example, the variable bit rate transport stream is encoded by the following sequence: First, the transport rate is set to the value of  $\text{TS\_recording\_rate}$ . The video stream is encoded with a variable bit rate. The transport packet is intermittently encoded by not employing null packets.

The 32-bit field of  $\text{RSPN\_arrival\_time\_discontinuity}$  is a relative address of a site where arrival timebase discontinuities are produced on the Bridge-Clip AV stream file. The  $\text{RSPN\_arrival\_time\_discontinuity}$  is sized with the source packet number as a unit and is counted with the value of  $\text{offset\_SPN}$  defined in the  $\text{ClipInfo}()$  as from the first source packet of the Bridge-Clip AV stream file. An absolute address in the Bridge-Clip AV stream file is calculated based on the aforementioned equation:

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}.$$

The 144-bit field of  $\text{reserver\_for\_system\_use}$  is reserved for a system. If  $\text{is\_format\_identifier\_valid}$  flag is 1, it indicates that the field of  $\text{format\_identifier}$  is effective. If  $\text{is\_format\_identifier\_valid}$  flag is 0, it indicates that the  $\text{format\_identifier}$

field is valid. If `is_original_network_ID_valid` flag is 1, it indicates that the field of `is_transport_stream_ID-valid` is valid. If the flag `is_transport_stream_ID-valid` is 1, it indicates that the `transport_stream_ID` field is valid. If `is_servec_ID_valid` flag is 1, it indicates that the `servec_ID` field is valid.

If `is_country_code_valid` flag is 1, it indicates that the field `country_code` is valid. The 32-bit field of `format_identifier` indicates the value of `format_identifier` owned by a registration descriptor (defined in ISO/IEC13818-1) in the transport stream. The 16-bit field of `original_network_ID` indicates the value of the `original_network_ID` defined in the transport stream.

The 16-bit field in `servec_ID` denotes the value of `servec_ID` defined in the transport stream. The 24-bit field of `country_code` shows a country code defined by ISO3166. Each character code is encoded by ISO8859-1. For example, Japan is represented as "JPN" and is encoded to "0x4A 0x50 0x4E". The `stream_format_name` is 15 character codes of ISO-646 showing the name of a format organization affording stream definitions of transport streams. An invalid byte in this field has a value of '0xFF'.

`Format_identifier`, `original_network_ID`, `transport_stream_ID`, `servec_ID`, `country_code` and `stream_format_name` indicate service providers of transport streams. This allows to recognize encoding limitations on audio or video streams and stream definitions of private data streams other than audio video streams or SI (service information). These information can be used to check if the decoder is able to decode

the stream. If such decoding is possible, the information may be used to initialize the decoder system before starting the decoding.

STC\_Info is now explained. The time domain in the MPEG-2 transport stream not containing STC discontinuous points (discontinuous points of the system time base) is termed the STC\_sequence. In the Clip, STC\_sequence is specified by the value of STC\_sequence\_id. Figs.50A and 50B illustrate a continuous STC domain. The same STC values never appear in the same STC\_sequence, although the maximum time length of Clip is limited, as explained subsequently. Therefore, the same PTS values also never appear in the same STC\_sequence. If the AV stream contains N STC discontinuous points, where  $N > 0$ , the Clip system time base is split into (N+1) STC\_sequences.

STC\_Info stores the address of the site where STC discontinuities (system timebase discontinuities) are produced. As explained with reference to Fig.51, the RSPN\_STC\_start indicates the address and begins at a time point of arrival of the source packet referenced by the (k+1)st RSPN\_STC\_start and ends at a time point of arrival of the last source packet.

Fig.52 shows the syntax of the STC\_Info. Turning to the syntax of STC\_Info shown in Fig.52, version\_number is four character letters indicating the version number of STC\_Info(). The version\_number must be encoded to "0045" in accordance with ISO 646.

Length is a 32-bit unsigned integer indicating the number of bytes of

ProgramInfo in the syntax of zzzz.clip shown in Fig.45 is now explained with

Length is a 32-bit unsigned integer indicating the number of bytes of ProgramInfo() from directly at back of this length field to the end of program(info()). If CPI\_type of CPI() indicates the TU\_map type, this length field may be set to 0. If the CPI\_type of CPI() indicates EP\_map type, the number\_of\_programs must be of a value not less than 1.

An 8-bit unsigned integer of `number_of_program_sequences` denotes the number of `program_sequences` in the Clip. This value indicates the number of for-loops next following this field. If `program_sequence` in the Clip is not changed, 1 must be set in the number of `program_sequences`. A 32-bit field of `RSPN_program_sequence_start` is a relative address where the program sequence commences on the AV stream.

`RSPN_program_sequence_start` is sized with the source packet number as a unit and is counted with the value of `offset_SPN` defined in the `ClipInfo()` as from the first source packet of the AV stream file. An absolute address in the AV stream file is calculated by:

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}.$$

The values of `RSPN_program_sequence_start` in the for-loop syntax must appear in the rising order.

A 16-bit field of `PCR_PID` denotes the PID of the transport packet containing an effective PCR field effective for the `program_sequence`. An 8-bit field of `number_of_audios` indicates the number of for-loops containing `audio_stream_PID` and `AudioCodingInfo()`. A 16-bit field of `video_stream_PID` indicates the PID of the transport packet containing a video stream effective for its `program_sequence`. `VideoCodingInfo()`, next following this field, must explain the contents of the video stream referenced by its `video_stream_PID`.

A 16-bit field of `audio_stream_PID` indicates the PID of a transport packet

An 8-bit field of `audio_component_type` indicates an audio component type

The CPI (Characteristics Point Information) in the syntax of zzzzz.clip shown in Fig.45 is explained. The CPI is used for correlating the time information in the AV stream with the address in its file. The CPI is of two types, namely EP\_map and TU\_map. In Fig.63, if CPI\_type in CPI() is EP\_map, its CPI() contains EP\_map. In Fig.64, if CPI\_type in CPI() is TU\_map, its CPI() contains TU\_map. One AV stream has one EP\_map or one TU\_map. If the AV stream is an SESF transport stream, the corresponding Clip must own an EP\_map.

The EP\_map in the CPI syntax shown in Fig.65 is explained. There are two types of the EP\_map, that is EP\_map for a video stream and an EP\_map for an audio stream. The EP\_map\_type in the EP\_map differentiates between these EP\_map types. If the Clip contains one or more video streams, the EP\_map for the video stream must be used. If the Clip does not contain a video stream but contains one or more audio

The sub table termed EP\_map\_for\_one\_stream\_PID() is created from one audio stream transmitted by the transport packet having the same PID to another. If there exist plural audio streams in the Clip, EP\_map may contain plural EP\_map\_for\_one\_stream\_PID().

Fig.70 shows the EP\_map syntax. By way of explanation of the EP\_map syntax shown in Fig.70, the EP\_type is a 4-bit field and shows the EP\_map entry point type, as shown in Fig.71. The EP\_type shows the semantics of the data field next following this field. If Clip includes one or more video stream, the EP\_type must be set to 0 ('video'). Alternatively, if the Clip contains no video stream but contains one or more audio stream, then EP\_type must be set to 1 ('audio').

The 16-bit field of `number_of_stream_PIDs` indicates the number of times of loops of the for-loop having `number_of_stream_PIDs` in the `EP_map()` as a variable. The 16-bit field of `stream_PID(k)` indicates the PID of the transport packet transmitting the number `k` elementary stream (video or audio stream) referenced by `EP_map_for_one_stream_PID (num_EP_entries(k))`. If `EP_type` is 0 ('video'), its elementary stream must be a video stream. If `EP_type` is equal to 1 ('audio'), its

The semantics of the 32-bit field of `RSPN_EP_start` differs with the `EP_type` defined in `EP_map()`. If `EP_type` is equal to 0 ('video'), this field indicates the relative address of the source packet including the first byte of the sequence header of the access unit referenced by the `PTS_EP_start` in the AV stream. Alternatively, if `EP_type` is equal to 1 ('audio'), this field indicates the relative address of the source

It is noted that  $TU\_start\_time(k)$  has a precision of 45 kHz.

Fig.74 shows the syntax of TU\_map. By way of explanation of the TU\_map syntax shown in Fig.74, the 32-bit field of offset\_time gives an offset time relative to TU\_map\_time\_axis. This value indicates the offset time relative to the first time\_unit in the Clip. The offset\_time is of a size based on 45 kHz clock derived from the 27 MHz precision arrival time clocks as unit. If the AV stream is to be recorded as new Clip, offset\_time must be set to 0.

The 32-bit field of time\_unit\_size affords the size of the time\_unit, and is based on 45 kHz clocks, derived from the 27 MHz precision arrival time clocks, as unit. Preferably, time\_unit\_size is not longer than one second ( $\text{time\_unit\_size} \leq 45000$ ). The 32 bit field of number\_of\_time\_unit\_entries indicates the number of entries stored in TU\_map().

The 32-bit field of RSN\_time\_unit\_start indicates the relative address of a site in the AV stream at which begins each time\_unit. RSN\_time\_unit\_start is of a size based on the source packet number as unit and is counted with the value of offset\_SPN defined in ClipInfo() as from the first source packet of the AV stream file as an initial value. The absolute address in the AV stream file is calculated by

$$\text{SPN\_xxx} = \text{RSPN\_xxx} - \text{offset\_SPN}.$$

It is noted that the value of RSN\_time\_unit\_start in the for-loop of the syntax must appear in the rising order. If there is no source packet in the number (k+1) time\_unit, the number (k+1) RSN\_time\_unit\_start must be equal to the number k RSPN\_time\_unit\_start.

Fig.75 shows the ClipMark syntax. By way of explanation of the ClipMark syntax shown in Fig.75, the version\_number is four character letters indicating the version number of this ClipMark. The version\_number must be encoded in accordance with ISO 646 to "0045".

Mark\_time\_stamp is a 32-bit field and stores the time stamp indicating a pointer having a specified mark. The semantics of mark\_time\_stamp differs with CPI\_type in the PlayList(), as shown in Fig.77.

If `CPI_type` in `CPI()` indicates the `EP_map` type, this 8-bit field indicates the `STC_sequence_id` of the continuous `STC` domain where there is placed `mark_time_stamp`. If `CPI_type` in `CPI()` indicates `TU_map` type, this 8-bit field has no meaning but is set to 0. The 8-bit field of `Character_set` indicates the indicating method of character letters encoded in the `mark_name` field. The encoding method corresponds

The 8-bit field of name\_length indicates the byte length of the mark name shown in the mark\_name field. This mark\_name field indicates the mark name. The byte number corresponding to the number of the name\_length from left of this field is the effective character number and denotes the mark name. In the mark\_name field, the values next following these effective character letters may be arbitrary.

Fig.78 shows another syntax of ClipMark which takes the place of Fig.75. Fig.79 shows a typical table of mark\_type which takes the place of Fig.76 in such case. Reserved\_for\_maker\_ID is a 16-bit field indicating the maker ID of the maker which, if mark\_type indicates a value from 0xC0 to 0xFF, defines the mark\_type. The maker ID is specified by a DVR format licensor. Mark\_entry() is the information indicating the point specified to a mark point. The syntax will be explained subsequently in detail. Representative\_picture\_entry is the information indicating the point of the information representative to the mark shown by mark\_entry(). The syntax of representative\_picture\_entry will be explained subsequently in detail.

ClipMark is used in order to enable a user reproducing an AV stream to retrieve the contents visually. A DVR player uses a GUI (graphical user interface) to present the ClipMark information to the user. For visual display of the ClipMark information, a picture indicated by `representative_picture_entry()`, rather than the picture indicated by `mark_entry()`, is to be indicated.

Fig.80 shows examples of `mark_entry()` and `representative_picture_entry`. It is assumed that a program name (title) of a certain program is displayed after some time, such as several seconds, as from the start of such program. If ClipMark is to be created, `mark_entry()` may be put at a start point of the program, with the `representative_picture_entry()` then being put at a point of display of the program name (title) of the program.

If the picture of `representative_picture_entry` is displayed on the GUI and the user has specified the picture, the DVR player starts replay as from the point where `mark_entry` is put.

Fig.81 shows the syntax of `mark_entry()` and `representative_picture_entry()`.

`Mark_time_stamp` is a 32-bit field and stores a time stamp indicating a point specified by the mark and a time stamp indicating a point of a picture representing a mark specified by `mark_entry()`.

Fig.82 shows a typical syntax of `mark_entry()` and `representative_picture_entry()` in case of employing the address based information rather than employing the information of the time stamp base by the PTS.

Offset\_nm\_pictures is a 32-bit field and indicates the number of pictures of offset from the picture referenced by RSPN\_ref\_EP\_start to the picture indicated by the mark point in the display sequence. This number is counted from 0. In the case of the embodiment of Fig.83, offset\_nm\_pictures is 6.

In case of `mark_entry()`, `RSPN_mark_point` indicates the relative address of the source packet including the first byte of the access unit referenced by the mark in the AV stream. In case of `representative_picture_entry()`, `RSPN_mark_point` indicates the relative address of the source packet including the first byte of the encoded picture representative of the mark indicated by the `mark_entry()`.

RSPN mark point is of a size having the source packet number as a unit and

is counted using the value of `offset_SPN` defined in the Clip Information file from the first source packet of the AV stream file.

Referring to Fig.85, the relation between ClipMark and EP\_map is explained. In the present embodiment, it is assumed that EP\_map specifies I0, I1 and In as addresses of the entry point, and that the an I-picture continuing to the sequence header from these addresses is started. If ClipMark specifies M1 as an address of a certain mark, and if the picture beginning from this source packet is to be decodable, it suffices if data readout is initiated from I1 as an entry point ahead of and closest to the M1 address.

MakerPrivateData has already been explained with reference to Fig.22 and hence is not explained here specifically.

Next, thumbnail\_information is explained. A thumbnail picture is stored in a menu.thmb file or in a mark.thmb file. These files are of the same syntax structure and own a sole Thumbnail(). The menu.thmb file stores a picture representing respective PlayLists. The totality of menu thumbnails are stored in the sole menu.thmb file.

The mark.thmb file stores a mark thumbnail picture, that is a picture representing a mark point. The totality of mark thumbnails corresponding to the totality of PlayLists and Clips are stored in the sole mark.thmb file. Since the thumbnails are frequently added or deleted, the operation of addition and partial deletion must be executable readily and speedily. For this reason, Thumbnail() has a block structure. Picture data is divided into plural portions each of which is stored in one tn\_block. One picture data

**Tn\_block\_size** is a 16-bit unsigned integer affording one tn\_block size with 1024 bytes as unit. For example, if **tn\_block\_size** = 1, it indicates that the size of one **tn\_block** is 1024 bytes. **Number\_of\_tn\_blocks** is a 16-bit unsigned integer representing the number of entries of **tn\_blocks** in the **Thumbnail()**. **Thumbnail\_index** is a 16-bit unsigned integer representing the index number of a thumbnail picture represented by the thumbnail information corresponding to one for-loop beginning from this

Figs.89A and 89B schematically show how thumbnail picture data are stored in `tn_block`. As shown in Figs.89A and 89B, the respective thumbnail picture data begin

One Aligned unit is made up of 32 source packets. The last Aligned unit in the DVR MPEG-2 transport stream is also made up of 32 source packets. So, the DVR MPEG-2 transport stream ends at a boundary of the Aligned unit. If the number of the transport packets of the input transport stream recorded on the disc is not a multiple of 32, the source packet having a null packet (transport packet of PID = 0x1FFF) must be used as the last Aligned unit. The file system is not allowed to add redundant



If the input transport stream is an SESF transport stream, Rpk must be equal to TS\_recording\_rate as defined in ClipInfo() associated with the AV stream file. If the input transport stream is not an SESF transport stream, the values defined in descriptors of the MPEG-2 transport stream, such as smoothing\_buffer\_descriptor, short smoothing bugger descriptor or partial\_transport\_stream\_descriptor, may be

By way of explaining the output timing of the MPEG-2 transport stream, if

arrival\_time\_stamp of the current source packet is equal to the value of the 30 LSB bits of arrival\_time\_clock(i), the transport packet of the source packet is extracted from the smoothing buffer 64. Ppk is an instantaneous maximum value of the transport packet rate. The smoothing buffer 64 is not allowed to overflow.

The parameters of the player model of the DVR MPEG-2 transport stream are the same as the parameters of the recorder model of the above-described DVR MPEG-2 transport stream.

Fig.93 shows the syntax of the Source packets, while transport\_packet() is an MPEG-2 transport packet as prescribed in ISO/IEC 13818-1. Fig.94 shows a TP\_extra\_header in the syntax of the source packet shown in Fig.93. By way of explaining the syntax of the TP\_extra\_header shown in Fig.94, copy\_permission\_indicator is an integer representing copy limitations of the payload of the transport packet. The copy limitations may be set to copy free, no more copy, copy once or copy prohibited. Fig.95 shows the relation between the values of copy\_permission\_indicator and the modes these values represent.

Copy\_permission\_indicator is appended to the totality of transport packets. If an input transport stream is to be recorded using the IEEE1394 digital interface, the value of copy\_permission\_indicator may be correlated with the value of EMI (encryption mode indicator) in the IEEE1394 isochronous packet header. If the input transport stream is recorded without using the IEEE1394 digital interface, the value of copy\_permission\_indicator may be correlated with the value of the CCI embedded in

In defining the Bridge-Clip AV stream, the Bridge-Clip AV stream must have a structure of a DVR MPEG-2 transport stream defined as described above. The Bridge-Clip AV stream must include one arrival time base discontinuous point. The transport stream ahead and at back of the arrival time base discontinuous point must

obey the limitations on encoding as later explained and also must obey the DVR-STD as later explained.

The present embodiment provides for seamless support of the seamless connection of the video and audio between PlayItems in editing. This seamless connection between the PlayItems guarantees the "continuous data supply" to the player/recorder and "seamless decoding". The "continuous data supply" means the capability of a file system in assuring data supply to a decoder at a bitrate necessary for prohibiting buffer underflow. The "continuous data supply" assures data storage continuously in terms of a block of a sufficient size to assure real-time properties and data readout from the disc as a unit.

The "seamless decoding" means the capability of the player in demonstrating audio/video data recorded on the disc without producing pause or gap in the decoder replay output.

The AV stream referenced by the seamlessly connected PlayItem is explained. Whether or not the connection between the previous PlayItem and the current PlayItem is guaranteed to enable seamless display can be verified from the connection\_condition field defined in the current PlayItem. There are a method for seamless connection between PlayItems employing Bridge-Clip and a method for seamless connection between PlayItems not employing Bridge-Clip.

Fig.96 shows the relation between the previous PlayItem and the current PlayItem. Fig.96 shows TSI made up of a shaded stream data of Clip1 (Clip AV

Fig.97 shows the relation between the previous PlayItem and the current PlayItem in case of not employing Bridge-Clip. In this case, the stream data read out from the player is shown shaded. In Fig.97, TS1 the shaded stream data of TS1 is data beginning from the address of a stream necessary for decoding the presentation unit associated with IN time of previous PlayItem (shown with IN\_time1 in Fig.97) and

extending up to the last source packet of Clip1. On the other hand, TS2 in Fig.97 is comprised of shaded stream data of Clip2 (Clip AV stream).

The shaded stream data of Clip2 of TS2 is stream data beginning at a first source packet of Clip2 and ending at an address of a stream necessary for decoding the presentation unit associated with the OUT\_time of the current PlayItem (shown OUT\_time2 in Fig.97).

In Figs.96 and 97, TS1 and T2 are continuous streams of source packets. Next, stream prescriptions of TS1 and TS2 and connection conditions therebetween are considered. As a limitation of the encoding structure of the transport stream, the number of video streams contained in TS1 and TS2 must be 1. The number of audio streams contained in TS1 and TS2 must be 2 or less. The numbers of audio streams contained in TS1 and TS2 must be equal to each other. It is noted that TS1 and/or TS2 may contain an elementary stream or a private stream other than those mentioned above.

Limitations on a video bitstream are now explained. Fig.98 shows an instance of seamless connection shown in the picture display sequence. In order that a video stream may be represented seamlessly, unneeded pictures represented at back of OUT\_time (OUT\_time of Clip1) and ahead of the IN\_time2 (IN\_time of Clip2) must be removed by a process of re-encoding a partial stream of the Clip near a junction point.

Fig.99 shows an instance of realizing seamless connection using

BridgeSequence in the case shown in Fig.98. The video stream of Bridge-Clip previous to RSPN\_arrival\_time\_discontinuity is made up of encoded video streams up to the picture associated with the OUT\_time1 of Clip1 of Fig.98. This video stream is connected to the video stream of the previous Clip1 and is re-encoded so as to prove one continuous elementary stream conforming to the MPEG2 standard.

In similar manner, the video stream of Bridge-Clip subsequent to RSPN\_arrival\_time\_discontinuity is made up of an encoded video stream as from a picture corresponding to IN\_time2 of Clip2 of Fig.98. The video stream can start decoding correctly, is connected to the video stream of the next following Clip2 and is re-encoded so as to become one continuous elementary stream conforming to MPEG2 standard. For creating Bridge\_Clip, several pictures in general need to be re-encoded, while other pictures can be copied from the original clip.

Fig.100 shows a case of realizing seamless connection without using BridgeSequence in the case of the embodiment shown in Fig.98. The video stream of Clip1 is made up of an encoded video stream up to a picture associated with OUT\_time1 of Fig.98, this being re-encoded so as to become one continuous elementary stream conforming to MPEG2 standard. Similarly, the video stream of Clip2 is comprised of an encoded video stream subsequent to the picture associated with the IN-time2 of Clip2 of Fig.98, this being re-encoded so as to become one continuous elementary stream conforming to MPEG2 standard.

By way of explaining encoding limitation of the video stream, the frame rate of

TS1 and TS2 video streams must be equal to each other. The TS1 video stream must end in `sequence_end_code`, while the TS2 video stream must start in `sequence_header`, `GOP header` and in an I-picture. The TS2 video stream must start in closed GOP.

The video presentation unit (frame or field) as defined in a bitstream must be continuous with a junction point in-between. At the junction point, the top/bottom field sequence must be continuous. In the case of encoding employing the 3-2 pulldown, it may be necessary to rewrite "top\_field\_first" and "repeat\_first\_field", while local re-encoding may also be made to prevent occurrence of a field gap.

By way of explaining encoding limitations on an audio bitstream, the audio sampling frequency of TS1 and that of TS2 must be equal to each other. The audio encoding method, such as MPEG1 layer 2, AC-3, SESF, LPCM and AAC of TS1 and TS2 must be equal to each other.

By way of explaining coding limitations on the MPEG-2 transport stream, the last audio frame of the TS1 audio stream must contain an audio sample having a display time equal to that at display end time of the display picture of TS1. The first audio frame of TS2 audio stream must contain an audio sample equal to that at display start time of the first display picture of TS2.

At a junction point, there must not be a gap in the sequence of the audio presentation unit. As shown in Fig.101, there may be an overlap defined by the length of the audio presentation unit not larger than two audio frame domain. The first packet

transmitting the TS2 elementary stream must be a video packet. The transport stream at the junction point must obey the DVR-STD which will be explained subsequently.

By way of explaining the limitations of Clip and Bridge-Clip, TS1 and TS2 must not contain discontinuous points of the arrival time base therein.

The following limitations are applied only to the case of using Bridge-Clip. The Bridge\_Clip AV stream has only one arrival time base discontinuous point at a junction point between the last source packet of TS1 and the first source packet of TS2. The RSPN\_arrival\_time\_discontinuity as defined in Clip Info() must indicate an address of the discontinuous point which in turn must indicate the address referencing the first source packet of TS2.

The source packet referenced by the RSPN\_exit\_from\_previous\_Clip as defined in BridgeSequenceInfo() may be any source packet in the Clip. It is unnecessary for this source packet to be a boundary of the Aligned unit. The source packet referenced by the RSPN\_exit\_from\_current\_Clip as defined in BridgeSequenceInfo() may be any source packet in the Clip. It is unnecessary for this source packet to be a boundary of the Aligned unit.

By way of explaining limitations on the PlayItem, the OUT\_time of previous PlayItem (OUT\_time 1 shown in Figs.96 and 97) must indicate the display end time of the last presentation unit of TS1. The IN\_time of the current PlayItem (IN\_time2 shown in Figs.96 and 97) must indicate the display start time of the first video presentation unit of TS2.

The first stream portion of Clip1(Clip AV stream file) must be arranged in a continuous area not shorter than one half fragment. The first stream portion of Clip2

(AV stream file) must be arranged in a continuous area not smaller than one half fragment.

Next, the DVR-STD is explained. The DVR-STD is a conceptual model for modeling the decoding processing in generating and verifying the DVR MPEG-2 transport stream. On the other hand, the DVR-STD is a conceptual model for modeling the decoding processing in generating and verifying the AV stream referenced by two PlayItems connected seamlessly as described above.

Fig.104 shows a DVR-STD model. The model shown in Fig.104 contains, as a constituent element, a DVR MPEG-2 transport stream player model. The notation such as  $n$ ,  $Tbn$ ,  $Mbn$ ,  $Ebn$ ,  $Tbsys$ ,  $Bsys$ ,  $Rxn$ ,  $Rbxn$ ,  $Rxsys$ ,  $Dn$ ,  $Dsys$ ,  $On$  and  $Pn(k)$  are the same as those defined in T-STD of ISO/IEC138188-1. That is,  $n$  is an index number of an elementary stream and  $Tbn$  is a transport buffer of an elementary stream  $n$ .

$Mbn$  is a multiplex buffer of the elementary stream  $n$ , and is present only in a video stream.  $Ebn$  is an elementary stream buffer in the elementary stream  $n$ . It is present only for the video stream.  $Tbsys$  is an input buffer for the system information in a program being decoded.  $Bsys$  is a main buffer in a system target decoder for the system information for a program being decoded.  $Rbxn$  is a transmission rate with which data is removed from  $Mbn$  and is present only for a video stream.

$Rxsys$  is a transmission rate with which data is removed from  $Tbsys$ .  $Dn$  is a decoder in an elementary stream  $n$ .  $Dsys$  is a decoder pertinent to the system information of a program being decoded.  $On$  is a re-ordering buffer in a video stream.

$P_n(k)$  is a number  $k$  presentation unit in the elementary stream  $n$ .

The decoding process for DVR-STD is explained. During the time a sole DVR MPEG-2 transport stream is being reproduced, the timing the transport packet is input to the Tbsys buffer is determined by the arrival\_time\_stamp of the source packet. The prescriptions for the decoding operation and the display operation are the same as those prescribed in ISO/IEC 13818-1.

The decoding process while the seamlessly connected PlayItem is being reproduced is explained. Here, reproduction of two AV streams, referenced by the seamlessly connected PlayItem, is explained. In the following explanation, the reproduction of the aforementioned TS1 And TS2, shown for example in Fig.96, is explained. Meanwhile, TS1 and TS2 are a previous stream and a current stream, respectively.

Fig.105 shows a timing diagram for inputting, decoding and displaying a transport packet when transferring from a given AV stream (TS1) to the next AV stream (TS2) seamlessly connected thereto. During the time of transferring from a preset AV stream (TS1) to the next AV stream (TS2), connected seamlessly thereto, the time axis of the arrival time base of TS2 (indicated STC2 in Fig.105) is not the same as the time axis of the arrival time base of TS1 (indicated STC1 in Fig.105).

Moreover, the time axis of the system time base of TS2 (indicated STC2 in Fig.105) is not the same as the time axis of the system time base of TS1 (indicated STC1 in Fig.105). Video display is required to be seamlessly contiguous, while there

may be overlap in the display time of the audio presentation unit.

The input timing to the DVR-STD is explained. Until the time T1, that is the time until the last video packet of TS1 is completely input to the TB1 of DVR-STD, the input timing to the TB1, Tbn or Tbsys of DVR-STD is determined by the arrival\_time\_stamp of the source packet of TS1.

The remaining packet of TS1 must be input to the buffer of Tbn or Tbsys at a bitrate of TS\_recording\_rate (TS1), where TS\_recording\_rate (TS1) is a value of the TS\_recording\_rate defined in the ClipInfo() associated with the Clip1. The time the last byte of TS1 is input to the buffer is the time T2. Thus, in the domain from time T1 until time T2, arrival\_time\_stamp of the source packet is discounted.

If N1 is the number of bytes of the transport packet next following the last video packet of TS1, the time DT1 from time T1 until time DT1 is the time required for N1 bytes to be completely input at a bitrate of TS\_recording\_rate (TS1), and is calculated in accordance with the following equation:

$$DT1 = T2 - T1 = N1 / \text{TS\_recording\_rate (TS1)}.$$

During the time as from T1 until T2, the values of Rxn and Rxsys are both changed to the value of TS\_recording\_rate (TS1). The buffering operation other than this rule are the same as that of the T-STD.

At time T2, the arrival time clock counter is reset to the value of arrival\_time\_stamp of the first source packet of TS2. The input timing of the TB1, Tbn or Tbsys to the DVR-STD is determined by the arrival\_time\_stamp of the source

packet of TS2. The Rxn and Rxsys are both changed to values defined in T-STD.

By way of explaining additional audio buffering and system data buffering, the audio decoder and the system decoder are required to have a surplus buffer quantity (approximately one second equivalent data volume) in addition to the buffer volume defined in T-STD.

By way of explaining the video presentation timing, display of the video presentation unit must be contiguous, via junction, in a manner free of gap. It is noted that STC1 is the time axis of the TS1 system time base (shown as STC1 in Fig.105), while STC2 is the time axis of the TS2 system time base (shown as STC2 in Fig.97). More accurately, STC2 begins at a time point the first PCR is input to T-STD.

The offset between STC1 and STC2 is determined as follows: If PTS1end is a PTS on STC1 associated with the last video presentation unit of TS1, and PTS2start is PTS on STC2 associated with the first video presentation unit of TS2, while Tpp is the display period of the last video presentation unit of TS1, the offset STC\_delta between two system time bases may be calculated in accordance with the following equation:

$$\text{STC\_delta} = \text{PTS1end} + \text{Tpp} - \text{PTS2start}.$$

By way of explaining the audio presentation, there may be overlap of display timing of audio presentation unit, which overlap may be 0 to less than 2 audio frames (see "audio overlap" shown in Fig.105). Which audio sample should be selected and re-synchronization of the display of the audio presentation unit to corrected time base

Next, on the time axis of the system time base in which STC1 and STC2 are converted on the same time axis, input of video packets from TS1 and the next

following video packet input from TS2 must not overflow nor underflow the video buffer.

Based on this syntax, data structure and rule, contents of data recorded on the recording medium or the replay information can be managed properly such that the user is able to confirm the contents of data properly recorded on the recording medium or reproduce desired data readily.

Although the present embodiment is explained taking the MPEG-2 transport stream as a multiplexed stream as an example, this is merely exemplary such that the present embodiment may be applied to a MPEG2 program stream or to a DSS transport stream used in DirecTV service (trademark) of USA.

The processing of locating reproduction of a scene represented by a mark point in case the syntax of `mark_entry()` and `representative_picture_entry()` is configured as shown in Fig.81 is explained with reference to the flowchart of Fig.106.

First, at step S1, a controller 23 of a recording and/or reproducing apparatus 1 reads out the EP\_map (Fig.70), STC\_Info (Fig.52) and ClipMark (Fig.78) as database of the DVR transport stream from the recording medium 100.

At step S2, the controller 23 formulates a list of a thumbnail from the picture referenced by `representative_picture_entry` (Fig.81) or `ref_thumbnail_index` to output the so formed list from a terminal 24 as a user interface input/output for display on a menu of GUI. If, in this case, `ref_thumbnail_index` has an effective value, `ref_thumbnail_index` is prioritized with respect to `representative_picture_entry`.

At step S3, a user designates a mark point of the reproduction start point. This is realized by the user selecting a thumbnail picture from the menu screen displayed as GUI. The controller 23 is responsive to this selecting operation to acquire a mark point associated with the designated thumbnail.

At step S4, the controller 23 acquires PTS of mark\_Time\_stamp and STC\_sequenc\_id of mark\_entry (Fig.81) specified at step S3.

At step S5, the controller 23 acquires, from STC\_Info (Fig.52), a source packet number at which to start the STC time axis associated with STC\_sequence\_id acquired at step S4.

At step S6, the controller 23 acquires, from the packet number at which starts the STC time axis acquired at step S5 and from a PTS of the mark point acquired at step S4, a source packet number where there is the entry point (I-picture) temporally previous and closest to the PTS of the mark point.

At step S7, the controller 23 reads out data of the transport stream from the source packet number containing the entry point acquired at step S6 to send the read-out data to the AV recorder 27.

At step S8, the controller 23 controls the AV recorder 27 to start the display as from the picture of the PTS of the mark point acquired at step S4.

The above-described operation is further explained with reference to Figs.107 to 109.

It is assumed that a DV transport stream has CM (commercial) inserted in the

DVR transport stream file has a STC time axis of STC\_EP\_start of STC\_sequence\_id = id = id0, as shown in Fig.107, and that the source packet number with which the time axis begins is smaller than the source packet number of the source beginning point A. It is also assumed that the CM (commercial) is added between the source packet numbers B and C.

In this case, there are registered in EP\_map, corresponding to the EP\_map shown in Fig.70, in association with A, B and C shown as RSPN\_EP\_start, respective PTSs, as PTS(A), PTS(B) and PTS(C), as shown in Fig.108.

Referring to Fig.109, there are recorded mark\_entry and representative\_picture\_entry, in the ClipMark of Fig.78, in association with the mark type values (Fig.79) of 0x92, 0x94 and 0x95 representing scene start, CM start and CM end, as shown in Fig.109.

As the Mark\_time\_stamp of mark\_entry, PTS(a1), PTS(b0) and PTS(c0) are registered in association with the scene start, CM start and CM end, with the respective STC\_sequence\_id being id0.

Similarly, as mark\_time\_stamp of the representative\_picture\_entry, PTS(a2), PTS(b0) and PTS(c) are registered in association with the scene start, CM start and CM end, with the respective STC\_sequence\_id being id0.

If  $PTS(A) < PTS(a1)$ , a packet number A is acquired at step S6. At step S7, a transport stream beginning at packet number A is supplied to the AV decoder 27 and, at step S8, display is initiated as from the PTS(a1).

Referring to the flowchart of Fig.110, the processing of CM skip reproduction in case the syntax of `mark_entry` and `representative_picture_entry` is configured as shown in Fig.81.

At step S21, the controller 23 reads out from the recording medium 100 the `EP_map` (Fig.70), `STC_Info` (Fig.52), `Program_Info` (Fig.54) and `ClipMark` (Fig.78). At step S22, the user designates CM skip reproduction from a terminal 24 as a user interface input/output.

At step S23, the controller 23 acquires the PTS of the mark information as the CM start point (0x94) and associated `STC_sequence_id` (Fig.81).

At step S24, the controller 23 acquires, from `STC_Info` (Fig.52), a source packet number of the CM beginning and end points, where begins the STC time axis corresponding to `STC-sequence-id`.

At step S25, the controller 23 causes the transport stream to be read out from the recording medium to route it to the AV decoder 27 to start the decoding.

At step S26, the controller 23 verifies whether or not the current displayed picture is a picture of the PTS of the CM start point. If the current display picture is not the picture of the CM start time point, the controller proceeds to step S27 to continue displaying the picture. The processing then reverts to step S25 to repeat subsequent steps.

If, at step S26, the current displayed picture is verified to be a picture of the PTS of the CM start point, the processing transfers to step S28 where the controller 23

controls the AV decoder 27 to halt the decoding and display.

At step S29, the controller 23 acquires the packet number where begins the STC time axis associated with the STC\_sequence\_id of the CM end point, while acquiring, from the packet number and the PTS of the CM end point acquired by the processing at step S23, the source packet number where there is an entry point temporally previous and closest to the PTS of the point.

At step S30, the controller 23 controls the AV decoder 27 to reinitiate display from the picture of the PTS of the CM end point.

At step S31, the controller 23 controls the AV decoder 27 to reinitiate display from the picture of the PTS at the CM end point.

Referring to Figs.107 to 109, the above-described operation is explained further. The CM start time point and the CM end point are present in this embodiment on the common time axis of STC\_sequence-id = id0. The source packet number where begins the STC time axis is selected to be smaller than the source packet number A of the scene start point.

If the transport stream is decoded and, at step S26, the display time is found to be the PTS (b0), that is if the display time point is found to be CM start time point, display is halted by the AV decoder 27. If  $PTS(C) < PTS(c0)$ , decoding is started from the stream, beginning from the data of the packet number C at step S30. At step S31, display is re-started from the picture of PTS(c0).

This method can be applied not only to CM skip reproduction but also to skip

reproduction of a scene between two points specified by the ClipMark in general.

Referring to the flowchart of Fig.112, the CM locating reproduction processing indicated by the mark point, in case mark\_entry and representative\_picture\_entry are of the syntax structure shown in Fig.82, is explained.

At step S41, the controller 23 acquires the information of EP\_map (Fig.70), STC\_Info (Fig.52), Program\_Info (Fig.54) and ClipMark (Fig.78).

At step S42, the controller 23 generates a list of thumbnails from the picture referenced by the representative\_picture\_entry (Fig.82) or the picture referenced by ref\_thumbnail\_index contained in the ClipMark (Fig.78), read out at step S41, to display the so generated list on a menu screen. If the ref\_thumbnail\_index has an effective value, ref\_thumbnail\_index is prioritized with respect to representative\_picture\_entry.

At step S43, the user specifies a mark point of the replay start point. This designation is carried out e.g., by the user selecting the thumbnail picture from the menu screen displayed by the processing of step S42 to specify the mark point associated with the thumbnail.

At step S44, the controller 23 acquires the RSPN\_ref\_EP\_start and offset\_num\_pictures (Fig.82) of the mark point specified by the processing of step S43.

At step S45, the controller 23 causes data of a transport stream to be read out from the source packet number corresponding to the RSPN\_ref\_EP\_start acquired at step S44 to route it to AV decoder 27.

If locating reproduction is commanded from a picture corresponding to scene start, the decoding is started from a stream beginning from data of a packet number A to count up pictures to be displayed, as from the picture PTS(A), without displaying, to start displaying as from a picture for which offset\_num\_pictures has assumed the value of M1.

The processing of CM skip reproduction in case the syntax of mark\_entry and representative\_picture\_entry are of a structure shown in Fig.82 is now explained by referring to the flowchart of Fig.116.

At step S61, the controller 23 acquires the information on EP\_map (Fig.70), STC\_Info (Fig.52), Program\_Info (Fig.54) and ClipMark (Fig.78).

If the user commands CM skip reproduction at step S62, the controller 23 acquires, at step S63, RSPN\_ref\_EP\_start and offset\_num\_pictures (Fig.82) as the mark information of respective points as CM start point and CM end point. The CM start point data is RSPN\_ref\_EP\_start(1) and offset\_num\_pictures(1), whilst the CM end point data is RSPN\_ref\_EP\_start(2) and offset\_num\_pictures(2).

At step S64, the controller 23 acquires PTS corresponding to RSPN\_ref\_EP\_start(1) and RSPN\_ref\_EP\_start(2) from EP\_map (Fig.70).

At step S65, the controller 23 causes a transport stream to be read out from the recording medium 100 to send it to the AV decoder 27.

At step S66, the controller 23 verifies whether or not the current display picture is the picture of PTS associated with RSPN\_ref\_EP\_start(1) and RSPN\_ref\_EP\_start(2) and, if the current picture displayed is not the picture of PTS associated with RSPN\_ref\_EP\_start(1), it proceeds to step S67 to cause the picture to be displayed continuously. The controller 23 then reverts to step S65 to repeat the subsequent process.

If, at step S66, the current picture displayed is the picture of PTS associated with

RSPN\_ref\_EP\_start(1), the controller 23 proceeds to step S68 to control the AV decoder 27 to count up the picture displayed as from the picture of the PTS associated with RSPN\_ref\_EP\_start(1) to halt the display when the count value is offset\_num\_pictures(1).

At step S69, the controller 23 causes transport stream to be read out beginning from the source packet number of RSPN\_ref\_EP\_start(2) to route the read-out data to the AV decoder 27.

At step S70, the controller 23 counts up the picture to be displayed, as from the picture of PTS associated with the RSPN\_ref\_EP\_start(2) (without displaying) and, when the count value is offset\_num\_pictures(2), the controller causes display to be started at the corresponding picture.

By way of explaining the above-described operation further by referring to Figs.113 to 115, the time PTS(B) and PTS(C) corresponding to the packet numbers B and C, respectively, are first obtained, based on EP\_map (Fig.114). The Clip AV stream is decoded and, when the value is N1 (Fig.15), the display is halted.

Decoding is re-started as from the stream beginning from the data of the packet number C to count up the picture displayed, as from the picture of the PTS(C) and, when the value is N2(Fig.115), display is re-started as from the picture.

The above processing may be applied not only to a case of CM skipping reproduction but also to reproduction of a scene with skipping between two points specified by the ClipMark.

The processing of locating a scene specified by a mark point and reproducing the so located scene is explained with reference to the flowchart of Fig.118 in case the syntax of mark\_entry and representative\_picture\_entry elementary stream is as shown in Fig.84 .

At step S81, the information on EP\_map (Fig.70), STC\_Info (Fig.52), Program\_Info (Fig.54) and ClipMark (Fig.78) is acquired.

At step S82, the controller 23 generates a list of thumbnails, referenced by representative\_picture\_entry or ref\_thumbnail\_index of ClipMark (Fig.78) to display the so generated list as GUI menu picture. If the ref\_thumbnail\_index has an effective value, ref\_thumbnail\_index is prioritized with respect to representative\_picture\_entry.

At step S83, a user specifies a mark point of the reproduction start point. This designation is made as the user selects a thumbnail picture from the menu picture and specifies the mark point associated with the thumbnail.

At step S84, the controller 23 acquires the RSPN\_mark\_point (Fig.84) specified by the user.

At step S85, the controller 23 acquires, from the EP\_map, the source packet number of the entry point previous and closest to the RSPN\_mark\_point of the mark\_point (Fig.70).

At step S86, the controller 23 reads out data of the transport stream from the source packet number associated with the entry point acquired at step S85 to route the so read out data to the AV decoder 27.

At step S87, the controller 23 controls the AV decoder 27 to start display as from the picture referenced by RSPN\_mark\_point.

The above-described processing is explained in more detail by referring to Figs.119 to 121. In the present embodiment, the DVR transport stream file is such a one in which a scene starts with the source packet and CM is inserted as from the source packet number B to the source packet number C. So, PTS\_EP\_start are registered as PTS(A), PTS(B) and PTS(C), in association with A, B and C as RSPN\_EP\_start of Fig.120. In the ClipMark shown in Fig.21, a1, b1 and c1 are registered as RSPN\_mark\_point of maker entry, while a2, b1 and c1 are registered as RSPN\_mark\_point of representative\_picture\_entry, in association with the scene start, CM start and CM end, respectively.

If, in locating a picture at a scene start for reproduction, the packet number A is such that  $A < a1$ , decoding is started as from a stream beginning from the data of the packet number A, such that display is started as from a picture corresponding to the source packet number a1.

The CM skipping reproduction processing in case the syntax of mark\_entry and representative\_picture\_entry is as shown in Fig.84 is now explained with reference to the flowcharts of Figs.122 and 123.

At step S101, the controller 23 acquires the information on EP\_map (Fig.70), STC\_Info (Fig.52), Program\_Info (Fig.54) and ClipMark (Fig.78).

At step S102, the user specifies CM skipping reproduction.

At step S103, the controller 23 acquires the RSPN\_mark\_point (Fig.84) of the mark information of each point for which the mark type (Fig.79) is the CM start point or the CM end point. The controller 23 sets the data of the CM start point and data of the CM end point as RSPN\_mark\_point (1) and as RSPN\_mark\_point(2), respectively.

At step S104, the controller 23 causes a transport stream to be read out from the recording medium 100 to output the read out stream for decoding.

At step S105, the controller 23 verifies whether or not the current picture is a picture corresponding to RSPN\_mark\_point (1). If the current picture is not a picture corresponding to RSPN\_mark\_point (1), the controller 23 proceeds to step S106 to continue to display the picture. The processing then reverts to step S11 to repeat the subsequent process.

If, at step S105, the current display picture has been verified to be a picture associated with RSPN\_mark\_point (1), the processing transfers to step S107, where the controller 23 controls the AV decoder 27 to halt the decoding and display.

Then, at step S108, the source packet number previous and closest to RSPN\_mark\_point (2) is acquired from EP\_map (Fig.70).

At step S109, the controller 23 reads out the transport stream data from the source packet number associated with the entry point acquired at step S108 to route it to the AV decoder 27.

At step S110, the controller 23 controls the AV decoder 27 to re-start the display as from the picture referenced by RSPN\_mark\_point (2).

By way of explaining the above-described processing in more detail by referring to the embodiment of Figs.119 to 121, the Clip AV stream is decoded and display is halted at a display picture corresponding to the source packet number b1 (Fig.21). If the source packet number  $C < \text{source packet number } c1$ , decoding is re-started as from the stream beginning from the data of the packet number C, and display is re-started as from the stream beginning from the data of the packet number c1.

As described above, a preset location is specified on the PlayList by a time stamp, which time stamp is converted in each Clip information of each Clip into a data address to have access to a preset position of the Clip AV stream.

More specifically, if a book mark or a Resume point is specified by the user as PlayList mark as time stamp on the time axis, the PlayList on reproduction may use the ClipMark of the ClipMark being referenced by the PlayList to access the scene start or end point of the Clip AV stream.

Meanwhile, the ClipMark syntax may be as shown in Fig.126 instead of in Fig.78.

In the present embodiment, RSPN\_mark is inserted in place of reserved\_for\_MakerID, mark\_entry() and representative\_picture\_entry of Fig.78. The 32-bit field of this RSPN\_mark denotes the relative address of the source packet containing the first byte of the access unit the mark is referencing. RSPN\_mark is sized in terms of the source packet number as a unit. It is defined in the Clip Information file as from the first source packet of the AV stream file and is counted with the value of

the offset\_SPN as an initial value.

The structure is otherwise the same as the structure of Fig.78.

The ClipMark syntax may further be configured as shown in Fig.127. In the present embodiment, RSPN\_ref\_EP\_start and offset\_num\_pictures are inserted in place of RSPN\_mark in Fig.126. These are similar to those shown in Fig.82.

Fig.128 shows an alternative syntax of ClipInfo().

Clip\_service\_type indicates the type of the AV stream file. For example, clip\_service\_type indicates types, such as video recording or audio recording. Moreover, clip\_service\_type may have the same meaning as that of the service type indicated by the digital TV broadcast program. For example, in digital BS broadcast in Japan, the service type has three types, namely the TV service, speech service and data broadcast service. The value representative of the service type of the program contained in an AV stream is set in the Clip\_service\_type.

Transcode\_mode\_flag is flag specifying the recording method for the MPEG-2 transport stream received from the digital broadcast. If 1 is set in this flag, it indicates that at least one elementary stream in the AV stream file corresponding to the Clip has been re-encoded and recorded. If this flag is set to 1, it indicates that the totality of the elementary streams are recorded without being changed from the contents as-received from the digital broadcast.

The other syntax fields have the same meaning as the fields bearing the same name as explained with reference to Fig.46.

Referring to Fig.129, another embodiment of the ProgramInfo() is explained.

A source packet sequence having constant program contents as provided for in the present format in the AV stream file is termed `program_sequence`.

An address in the AV stream file where starts a new `program_sequence` is stored in ProgramInfo(). This address is indicated by `SPN_program_sequence_start`.

The `program_sequence` other than the last `program_sequence` in the AV stream file begins with the source packet specified by its `SPN_program_sequence_start` and ends with the source packet directly previous to the source packet specified by the next `SPN_program_sequence_start`. The last `program_sequence` begins with the source packet specified by the `SPN_program_sequence_start` and ends with the last source packet of the AV stream file.

The `program_sequence` may be astride the boundary of the `STC_sequence`.

The length indicates the number of bytes as from the byte directly following the length field up to the last byte of ProgramInfo().

The `num_of_program_sequence` indicates the number of `program_sequences` in the AV stream file.

`SPN_program_sequence_start` indicates an address where begins the `program_sequence` on the AV stream file. `SPN_program_sequence_start` is sized with the source packet number as basis and is counted from the first source packet of the AV stream file, with 0 as an initial count.

The values of the `SPN_program_sequence_start`, entered in ProgramInfo(), are

arrayed in the rising order.

It is presupposed that SPN\_program\_sequence\_start is indicating the source packet with respect to the program\_sequence. SPN\_program\_sequence\_start is created by the recorder (recording and/or reproducing apparatus 1 of Fig.1) and analyzes the PSI/SI in the transport stream. Since delay time is required for the recorder to analyze the PSI/SI to detect the change therein, PN\_program\_sequence\_start may indicate the source packet within a preset time as from the actual PSI/SI change point.

Program\_map\_PID is a value of the PID of the transport packet having the PMT (program map table) applicable to the program\_sequence.

Num\_of\_streams\_in\_ps indicates the number of elementary streams defined in the program\_sequence thereof.

Num\_of\_groups indicates the number of the elementary streams defined in the program\_sequence thereof. The num\_of\_groups is a number not smaller than 1.

If the PSI/SI of the transport stream owns the group information of the elementary stream, the num\_of\_groups is presupposed to have a value not smaller than 1.

Stream\_PID indicates the value of PID for the elementary stream defined in the PMT referenced by program\_map\_PID of the program\_sequence thereof.

StreamCodingInfo() indicates the information of the elementary stream specified by the stream\_PIDm as will be explained in detail subsequently.

Num\_of\_streams\_in\_group indicates the number of elementary streams owned by the group of the elementary streams.

Stream\_index indicates the value of stream\_index defined in the order defined in a for-loop in the stream\_index of the syntax corresponding to the elementary stream owned by the group of the elementary streams.

Fig.30 shows a syntax of StreamCodingInfo().

Length indicates the number of bytes from a byte directly after the length field to the last byte of StreamCodingInfo().

Stream\_Coding\_type indicates the encoding type of the elementary stream specified by the stream\_PID corresponding to this StreamCodingInfo(). The meaning of the values is the same as that shown in Fig.131.

Video\_format indicates the video format of the video stream specified by the stream\_PID corresponding to this StreamCodingInfo().

The meaning of the values is the same as that shown in Fig.56.

Frame\_rate indicates the frame rate of the video stream specified by the stream\_PID corresponding to this StreamCodingInfo().

The meaning of the values is the same as that shown in Fig.57.

Display\_aspect\_ratio denotes the display aspect ratio of the video stream specified by the stream\_PID corresponding to this StreamCodingInfo().

The meaning of the values is the same as that shown in Fig.58.

Cc\_flag is a flag specifying whether or not a closed caption signal (closed

caption data) in the video stream specified by the stream\_PID corresponding to this StreamCodingInfo() has been encoded.

Original\_video\_format\_flag is a flag indicating whether or not there exist original\_video\_format and original\_aspect\_ratio in this StreamCodingInfo().

Original\_video\_format is an original video format previous to encoding of the video stream specified by stream\_PID corresponding to this StreamCodingInfo(). The meaning of the values is the same as video\_format described above with reference to Fig.58.

Original\_display\_aspect\_ratio is an original display aspect ratio prior to the encoding of the video stream specified by the stream\_PID corresponding to this StreamCodingInfo(). The meaning of the values is the same as that of display\_aspect\_ratio (Fig.58).

It is assumed that, in transcoding a transport stream in which a multi-media data stream, such as BML stream or title, is multiplexed along with the video stream, the video format is changed such as from 1080i to 480i by re-encoding the video stream, with the multi-media data stream keeping the contents of the original stream. On the other hand, it is assumed that the multi-media data stream maintains the contents of the original stream. There are occasions where information mismatching is produced between the new video stream and the multi-media data stream. For example, while the parameters pertinent to display of the multi-media data stream are determined as the video format of the original video stream is presumed, the video format may be

changed as a result of re-encoding the video stream. In such case, the information pertinent to the original video stream is saved in `Original_video_format` and in `original_display_aspect_ratio`. The reproducing device (recording and/or reproducing apparatus 1 of Fig.1) creates a display picture from the above-mentioned new video stream and from the multi-media data stream as follows:

- The video stream is up-sampled in accordance with the video format shown by `original_video_format` and in `original_display_aspect_ratio`.
- The upsampled picture and the multi-media data are synthesized to form a correct display picture.

`Audio_presentation_type` denotes the presentation type of the audio stream specified by `stream_PID` corresponding to this `StreamCodingInfo()`.

The meaning of the value is the same as that of `audio_component_type` of Fig.61.

`Sampling_frequency` means the sampling frequency of the audio stream indicated by `stream_PID` corresponding to this `StreamCodingInfo()`.

The meaning of the value is the same as that shown in Fig.62.

Another embodiment of `EP_map` is explained. In this embodiment of the `EP_map`, data of `PTS_EP_start` and `RSPN_EP_start`, explained with reference to Figs.70 and 72, but is stored in `EP_map` after compression encoding aimed to reduce the data volume. This compression encoding is executed by the controller 23 of Fig.1.

`EP_map` is comprised of at least one or more sub-table termed



EP\_map has one EP\_map\_for\_one\_stream\_PID for a continuous range of a stream transmitted in the same PID.

In the case of Fig.69, program#1 and program#3 own the same video PID, however, the data range is not continuous, so that each program must own EP\_map\_for\_one\_stream\_PID.

For reducing the data size of the EP\_map\_for\_one\_stream\_PID() table and improving data search performance, EP\_map\_for\_one\_stream\_PID() is divided into two subtables, namely EP\_coarse and EP\_fine (see Fig.132).

An EP\_fine entry owns the bit information on the side LSB (least significant bit) of PTS\_EP\_start and RSPN\_EP\_start (see Figs.133 and 134). The EP\_coarse entry has the bit information on the side MSB (most significant bit) of PTS\_EP\_start and RSPN\_EP\_start and the entry number in the table of EP\_fine associated therewith (entry in the EP\_fine table having the LSB side bit taken from the same PTS\_EP\_start).

The number of entries in the EP\_coarse subtable is significantly smaller than that of the EP\_fine subtable. The EP\_coarse entries are formed in the following cases:

- entry of first PTS\_EP-fine;
- entry of first RSPN\_EP\_fine after wraparound of the value of RSPN\_EP\_fine; and
- entry of first RSPN\_EP\_fine after wraparound of the value of RSPN\_EP\_fine (see Fig.135).

An instance of random accessing to an AV stream in case of using EP\_map is

hereinafter explained.

It is assumed that a PlayList is to be reproduced after lapse of 30 minutes on its global time axis.

- STC-sequence-id of the PlayItem containing the time corresponding to the time after 30 minutes on the global time axis of the STC-sequence.
- The value of PTS corresponding to the time after lapse of 30 minutes on the local time axis of the above-mentioned STC-sequence.
- The above-mentioned STC-sequence RSPN\_STC\_start is derived from STC\_Info.
- Data search is started from the entry in the EP\_subtable where RSPN\_EP\_coarse is not less than RSPN\_STC\_start. In the EP\_coarse subtable, an entry of PTS\_EP\_coarse having a value closest having a value temporally previous to the required PTS is found.
- In the EP\_fine subtable, data search is started from the entry number of EP\_fine associated with the PTS\_EP\_coarse thus found. Such an entry is found which is closest and has a value temporally previous to the PTS as required is found.
- The RSPN\_EP\_start for starting the readout of the access unit of the PTS as required is determined by substituting 18 LSB side bits of the RSPN\_EP\_coarse associated with the PTS\_EP\_coarse thus found for the RSPN\_EP\_fine bits corresponding to PTS\_EP\_fine thus found..

Fig.136 shows the syntax of EP\_map explained above.

Number\_of\_stream\_PID\_entries indicates the number of entries of EP\_map\_for\_one\_stream\_PID in the EP\_map.

Stream\_PID[k] indicates the value of PID of a transport packet transmitting the elementary stream referenced by the above-mentioned EP\_map\_for\_one\_stream\_PID which has an entry number k in EP\_map.

EP\_stream\_type[k] indicates the type of the elementary stream referenced by the map\_for\_one\_stream\_PID. The meaning of the value is indicated in Table of Fig.137.

If EP\_stream\_type[k] is 0 or 1, its elementary stream is a video stream. The meaning of video type 1 and the video type 2 are explained later in connection with explanation of EP\_video\_type (Fig.139).

If EP\_stream\_type[k] is 2, its elementary stream is an audio stream.

Num\_EP\_fine\_entries[k] indicates the number of EP-coarse entries in the EP\_map\_for\_one\_stream\_PID.

Num\_EP\_fine\_entries[k] indicates the number of P-fine entries in the EP\_map\_for\_one\_stream\_PID.

EP\_map\_for\_one\_stream\_PID\_start\_address[k] denotes a relative address position in the EP\_map() where begins the above-mentioned EP\_map\_for\_one\_stream\_PID. This value is indicated by the number of bytes as from the first byte of EP\_map.

Fig.138 shows the syntax for EP\_map\_for\_one\_stream\_PID. For explanation of this semantics, the meaning of the PTS\_EP\_start and RSPN\_EP\_start as the origin of data stored in the EP\_map\_for\_one\_stream\_PID is explained.

PTS\_EP\_start and RSPN\_EP\_start associated therewith indicate an entry point

on the AV stream. PTS\_EP\_fine and PTS\_EP\_coarse associated therewith are derived from the same RSPN\_EP\_start. PTS\_EP\_start and RSPN\_EP\_start are defined as follows:

EP\_start is defined as follows:

PTS\_EP\_start is a 33-bit long unsigned integer. The definition of PTS\_EP\_start differs with the value of EP\_stream\_type for EP\_map\_for\_one\_stream\_PID.

If EP\_stream\_type is 0 ('video type 1'), then PTS\_EP\_start indicates the 33-bit PTS of the video access unit beginning from the sequence header in the AV stream.

If EP\_stream\_type is 2 ('audio'), then PTS\_EP\_start indicates the 33-bit PTS of the video access unit defined beginning from the sequence header in the AV stream.

If EP\_stream\_type is 1 ('video type 2'), then PTS\_EP\_start indicates the 33-bit PTS of the video access unit defined in Fig.139 in accordance with the value of the EP\_video\_type associated therewith.

RSPN\_EP\_start is a 32-bit unsigned integer. The definition of RSPN\_EP\_start differs with the value of EP\_stream\_type for EP\_map\_for\_one\_stream\_PID.

If EP\_stream\_type is 0 ('video type 1'), this field indicates an address in the AV stream of a source packet containing a first byte of the video access unit associated with PTS\_EP\_start.

If EP\_stream\_type is 2 ('audio'), this field indicates an address in the AV stream of the source packet containing the first byte of the audio access unit associated with the PTS\_EP\_start.

If EP\_stream\_type is 1 ('video type 2'), the meaning of RSPN\_EP\_start is defined in Fig.139 in accordance with the value of EP\_video\_type associated therewith.

RSPN\_EP\_start is represented in terms of the source packet number as unit and is counted with 0 as an initial value as from the first source packet in the AV stream file.

The semantics of EP\_map\_for\_one\_stream\_PID are explained.

If EP\_fine\_table\_start\_address indicates the start address of the first byte of the first EP\_video\_type[EP\_fine\_id]. This address is represented by the number of relative bytes as from the first byte of EP\_map\_for\_one\_stream\_PID. The number of relative bytes begins from 0.

Ref\_to\_EP\_fine\_id denotes the number of an EP\_fine entry having PTS\_EP\_fine related with PTS\_EP\_coarse next following this field. PTS\_EP\_fine and PTS\_EP\_coarse associated therewith are derived from the same PTS\_EP\_start.

Ref\_to\_fine\_id is given by the value of EP\_fine\_id defined in the order stated in for-loop of EP\_fine\_id.

PTS\_EP\_coarse and PTS\_EP\_fine on one hand and RSPN\_EP\_coarse and RSPN\_EP\_fine on the other may be derived as follows:

There are Nf entries in the EP\_fine subtable, these entries being arrayed in the rising order of the values of the associated RSPN\_EP\_start.

The respective PTS\_EP\_fine entries may be derived from the respective PTS\_EP\_start as follows:

..entries of first RSPN EP fine after wraparound of the RSPN\_EP\_fine values.

Fig.140 shows a flowchart of the recording operation of the Clip AV stream file and the pertinent Clip Information file. Reference is made to the recording and/or reproducing apparatus of Fig.1 for explanation.

At step S201, the controller 23 forms a transport stream obtained on encoding an AV input from terminals 11 and 12 or a transport stream input from a digital interface of the terminal 13 into a file to create and record a Clip AV stream file.

At step S202, the controller 23 forms a ClipInfo on the AV stream file.

At step S203, the controller 23 forms a SYNC\_Info on the AV stream file.

At step S204, the controller 23 forms a Program\_Info on the AV stream file.

At step S205, the controller 23 forms a CPI (EP\_map or TU\_map) on the AV stream file.

At step S206, the controller 23 forms a ClipMark on the AV stream file.

At step S207, the controller 23 records a Clip Information file having stored therein ClipInfo, STC\_Info, ProgramInfo, CPI and ClipMark.

Although the respective processings are explained chronologically, steps S201 to S206 in effect operate simultaneously.

An illustrative operation for crating STC\_Info is explained using the flowchart of Fig.14.

At step S221, a stream analysis unit 18 checks whether or not a PCR packet has been received. If the result of check at step S221 is NO, processing reverts to step S221 and, if otherwise, processing transfers to step S222.

At step s222, it is checked whether or not STC discontinuities have been detected. If the result is NO, processing reverts to step S221 and, if otherwise, processing transfers to step S223. If the PCR packet is the first one received since the start of recording, processing necessarily transfers to step S222.

At step s223, the number (address) of the transport packet transporting the first PCR of a new STC is acquired.

At step S224, STC\_Info is prepared.

At step S225, it is checked whether or not the inputting of the last transport packet has been finished. If the result of check is NO, processing reverts to step S221 and, if otherwise, processing comes to as close.

A typical operation of preparing Program\_Info is explained using a flowchart of Fig.142. This processing is carried out by a multiplexing stream analysis unit 18 of Fig.1.

At step S141, the stream analysis unit 18 checks whether or not a transport packet inclusive of PSI/SI has been received. Specifically, the transport packet of PSI/SI is a packet of PAT, PMT or SIT. SIT is a transport packet stating the service information of a partial transport stream provided in DVB standard. If the result of check at step S241 is NO, processing reverts to step S241 and, if otherwise, processing transfers to step S242.

At step S242, it is checked whether or not the contents of PSI/SI have been changed. That is, it is checked whether or not the respective contents of PAT, PMT and

SIT have been changed from the respective contents previously received. If the contents have not been changed, processing transfers to step S243. Meanwhile, if the PSI/SI is the first one received since the start of recording, processing necessarily transfers to step S243.

At step S243, the number (address) and contents of the transport packet transmitting a new PSI/SI are acquired.

At step S244, the information on `program_sequence` is formed.

At step S245, it is checked whether or not the inputting of the last transport packet has been finished. If the result is NO, processing reverts to step S241 and, if otherwise, processing is terminated.

A typical operation of preparing `EP_map` is explained using the flowchart of Fig.143.

At step S261, the stream analysis unit 18 sets a PID of the video of the AV program to be recorded. If there are plural video contents in the transport stream, the respective video PDs are set.

At step S262, the stream analysis unit 18 receives a video transport packet.

At step S263, the stream analysis unit checks whether or not the payload of the transport packet (data part connecting to packet header) begins with the first byte of the PES packet (the PES packet is a packet provided in MPEG2 and packetizes an elementary stream). This may be grasped by checking for the value of "`payload_unit_start_indicator`" in the transport packet header. If this value is 1, the

payload of the transport packet begins with the first byte of the PES packet. If the result of check at step S263 is NO, processing reverts to step S262 and, if otherwise, processing transfers to step S264.

At step S264, the stream analysis unit checks whether or not the payload of the PES packet begins with the first byte of sequence\_header\_code of MPEG video (code of "0x000001B3 in 32 bits". If the result of check at step S264 is NO, processing reverts to step S262 and, if otherwise, processing transfers to step S265.

At step S265, the current transport packet is to be an entry point.

At step S266, the stream analysis unit acquires the packet number of the packet, PTS of an I-picture video PID beginning at the sequence\_header\_code and the video PID to which belongs its entry point for inputting to the controller 23. The controller 23 forms EP\_map.

At step S267, it is checked whether or not the current packet is the last input transport packet. If it is not the last packet, processing reverts to step S262 and, if otherwise, the processing is terminated.

Fig.144 shows a flowchart for explaining the method for creating ClipMark of the syntax of Fig.81 in encoding and recording analog AV signals.

At step S281, the analysis unit 14 analyzes the input AV signals from terminals 11, 12 to detect characteristic points specifying characteristic scenes ascribable to AV stream contents, such as program locating or scene change points.

At step S282, the controller 23 acquires PS of a picture of the characteristic

At step S305, the controller 23 stores the information of the characteristic point in the ClipMark. Specifically, the controller 23 stores the syntax of the ClipMark and the information explained in connection with the semantics.

The method for performing special reproduction using EP\_map is explained. EP\_map is useful in making random access reproduction.

In the transport stream of the digital broadcast, video PID is likely to be changed. So, the decoder has to be aware of PID mapping in the transport stream being decoded. So, the EP\_map has the value of the video PID from one subtable termed P\_map\_for\_one\_stream\_PID to another, while the ProgramInfo has the information on PID mapping.

Fig.146 shows a case of DVR MPEG2 TS in which the video PID value is changed in a transport stream. In this case, EP\_map has the information on PID mapping.

Fig.147 shows a player model in case of I-picture search (e.g., trick play or chapter search).

- 1) First, a file system 112 reads data of Clip Information file (EP\_map, STC\_Info and ProgramInfo) from a disc 111 (corresponding to the recording medium 100 of Fig.1). The data is sent to a host computer 15.
- 2) A user interface sets PTS of the search start time and the program number to be reproduced. The data is sent to the host computer 115.
- 3) The host computer 115 sets in demultiplexer 113 the video PID of a source packet specified by RSPN\_EP\_start corresponding to search start time.
- 4) The host controller 115 sets in the file system 112 the data address corresponding to the source packet specified by RSPN\_EP\_start corresponding to the search start

time.

- 5) The file system 112 reads out DVR MPEG-2 transport stream from the specified data address.
- 6) If the user sets the next search time, processing reverts to step S2.

Fig.148 shows an original AV stream file and the same AV stream file as edited for erasing a stream of a partial reproduction range of the stream.

It is assumed that, prior to editing, the VirtualPlayList is indicating the IN\_time and OUT\_time on the original AV stream. If an editing of erasing a stream portion not used by the Virtual PlayList (Minimizing editing) is made, it changes the original AV stream to an as-edited stream. Data from the leading end of the original AV stream to a point X and data from a point Y to the trailing end are erased. In the following explanation, a typical method for determining these X and Y points is explained.

Fig.149 illustrates the method for rasing unneeded data ahead of an IN point. PlayList denotes the point IN of the original AV stream. Fig.149 also shows EP\_map of the AV stream. For decoding the picture specified by the IN point, an I-picture beginning from the address ISA2 is needed.

Moreover, PAT, PMT and PCR packets are needed after point X. The PTS of RSPN\_EP\_start = 1 and PTS of SA1 is pts1, while PTS of RSPN\_EP\_start = ISA2 is pts2. If the time difference of the system time base of pts1 and that of pts2 is not less than 100 msec, there exist PAT, PMT and PCR packets between the address ISA1 and ISA2, at least insofar as the case of SESF, DVB, ATSC and ISDB is concerned.

So, the point X is determined before ISA1. The point X must be on a boundary of the aligned units.

A recording device ( recording and/or reproducing apparatus 1 of Fig.1) is able to decide the point X at the next step, using EP\_map, without analyzing the contents of the AV stream.

- 1) RSPN\_EP\_start having a value of PTS of the display time closest to and more past than the PTS of the IN time on the system time base is found out.
- 2) RSPN\_EP\_start having a value of PTS of the display time more past at least 100 msec than the value of the PTS of the RSPN\_EP\_start found at step 1) is found out.
- 3) The X point is determined before RSPN\_EP\_start as found at step 2). The point X must be on the boundary line of the aligned unit.

This method is simple because it is unnecessary to read out data of the AV stream in order to determine the X point to analyze its contents. However, there are occasions wherein, with the as-edited AV stream, unneeded data be left in the reproduction of the PlayList. If the data of the AV stream is read out to determine its contents in order to determine the point X, unneeded data can be erased efficiently in reproducing the PlayList.

Fig.150 illustrates a method for erasing unneeded data at back of the OUT point. PlayList specifies the OUT point on the original AV stream. There is also shown EP\_map of the AV stream.

The video sequence started from RSPN\_EP\_start = ISA4 is presupposed to be

I2 B0 B1 P5 ...

where I, P and B denote I-, P- and B-pictures, respectively. If, in this processing, the recorder fails to analyze the contents of the AV stream, the recorder is unable to realize the information of the picture the PTS of OUT\_time references, such as picture coding type or temporal reference. OUT\_time PTS may be referencing the picture B0 or the picture B1 (this cannot be clarified unless the recorder analyzes the contents of the AV stream). If the recorder is to decode the pictures B0 or B1, I2 is required. The PTS of I2 is larger than PTS of OUT\_time ( $\text{OUT\_time} < \text{pts4}$ , here pts4 is PTS of I2). Although PTS of I2 is larger than PTS of OUT\_time, I2 is required for B0 and b1.

Therefore, the point Y is determined so as to be at back of the address ISA5 shown. ISA5 is the value of RSPN\_EP\_start directly at back of ISA4 in EP\_map. The point Y must be on the boundary of the island unit.

The recorder is also able to decide the point Y at the next step, using EP\_map, without analyzing the contents of the AV stream.

- 1) RSPN\_EP\_start having the value of PTS of the display time point closet to and more future than the PTS of OUT\_time on the system time base is found.
- 2) RSPN\_EP\_start lying directly at back of RSPN\_EP\_start found at step 1) is found.
- 3) The point Y is determined so as to be more backward than RSPN\_EP\_start found at step 2).

The point Y must be on the boundary of the island unit.

This method is simple because it is unnecessary to read out data of the AV

stream in order to determine the Y point to analyze its contents. However, there are occasions wherein, with the as-edited AV stream, unneeded data be left in the reproduction of the PlayList. If the data of the AV stream is read out to determine the point Y to analyze its contents, unneeded data can be erased efficiently in reproducing the PlayList.

Based on the syntax, data structure and the rule, described above, the contents of data recorded on a recording medium, or the reproduction information, may be managed properly to enable the user to confirm the contents of data recorded on the recording medium during reproduction or to reproduce desired data extremely readily.

With the data base structure of the present invention, a PlayList file and the Clip Information file are recorded separately, so that, in case the contents of a given PlayList or Clip are changed by editing, it is unnecessary to change other irrelevant files. So, file contents can be changed readily, while the time needed in such change or recording may be shorter.

If initially the Info.dvr only is read out, the disc recording contents are presented to the user interface, and only the PlayList file commanded to be reproduced by the user and the relevant Clip Information file are read out from the disc, the user queuing time may be shorter.

If the totality of PlayList files or Clip Information file are collected into a sole file for recording, the file size becomes excessively large, so that the operation in changing and recording the file contents is considerably time-consuming as compared

Although the above-described sequence of operations can be executed by a hardware, it can also be executed by software. In such case, the recording and/or reproducing apparatus 1 is constituted by, for example, a personal computer shown in Fig.151.

In Fig.151, the CPU (central processing unit) executes a variety of operations in accordance with a program stored in a ROM (read-only memory) 202 or a program loaded from a memory 208 to a RAM (random access memory) 203. In the RAM 203, data necessary for the CPU 201 to execute a variety of operations are also stored as necessary.

The CPU 201, ROM 202 and the RAM 203 are interconnected over a bus 204, connected in turn to an input/output interface 205.

To the input/output interface 205 are also connected an input unit 206, such as a keyboard or a mouse, a display, such as CRT or LCD, an output unit 207, such as a loudspeaker, a memory 208, such as hard disc, and a communication unit 209, made up e.g., of a modem or a terminal adapter. The communication unit 209 performs communication processing over a network.

A driver 210 is connected as necessary to the input/output interface 205. There are also loaded a magnetic disc 221, an optical disc 222, a magneto-optical disc 223 or a semiconductor memory 224, and a computer program, read out therefrom, is installed as necessary on the memory 208.

The above-described sequence of operations may be executed not only on a hardware but also on a software. If the sequence of operations is to be carried out on the software, the program forming the software or a variety of programs are installed on a dedicated hardware of a computer, such that the programs are installed from a recording medium, such as a general-purpose personal computer.

Referring to Fig.151, this recording medium may be constituted by a package medium distributed for furnishing the user with a program, in addition to a computer, such as magnetic disc 221 (inclusive of floppy disc), an optical disc 222 (inclusive of CD-ROM (Compact Disc– Read Only memory) and DVD (Digital versatile Disc)), a magneto-optical disc 223 (inclusive of MD (Mini-Disc)) or a semiconductor memory 224. In addition, the recording medium is constituted by a hard disc furnished to the user as it is pre-loaded on a computer and which includes a ROM 202 or a memory 208 having the program stored therein.

In the present specification, the respective steps stating the sequence of the program furnished by a medium includes not only the processing executed chronologically in accordance with the stated order but also the processing executed in parallel or batch-wise.

The system in the present specification denotes an entire apparatus made up of plural devices.

### Industrial Applicability

According to the present invention, described above, the start address information of a domain in which the encoding information in the AV stream is continuous, the information for correlating the time information in the AV stream and the address information, and the time information of a characteristic picture in the AV stream, are recorded as the Clip information.

So, the operation of determining the readout position of the AV stream or the decoding operation can be performed promptly in any case and, in particular, preset marks can be retrieved promptly.